Reconstruction 3D et localisation simultanée de caméras mobiles : une approche temps-réel par ajustement de faisceaux local

Etienne Mouragnon

N° d'ordre : D.U. 1799
EDSPIC : 392

Université Blaise Pascal - Clermont-Ferrand II

École Doctorale
Sciences Pour l'Ingénieur de Clermont-Ferrand

Thèse présentée par :
Etienne Mouragnon

Formation Doctorale CSTI :
Composants et Systèmes pour le Traitement de l'Information

en vue de l'obtention du grade de

Docteur d'Université

spécialité : Vision pour la robotique

Reconstruction 3D et localisation simultanée de caméras mobiles : une approche temps-réel par ajustement de faisceaux local

Soutenue publiquement le 5 décembre 2007 devant le jury :

M. Jean-Marc Lavest	Président
M. François Chaumette	Rapporteur et examinateur
M. El Mustapha Mouaddib	Rapporteur et examinateur
M. Peter Sturm	Rapporteur et examinateur
M. Fabien Dekeyser	Examinateur
M. Maxime Lhuillier	Examinateur
M. Michel Dhome	Directeur de thèse

Remerciements

Je remercie tout d'abord François Chaumette, El Mustapha Mouaddib et Peter Sturm qui ont bien voulu faire partie de mon jury de thèse et examiner mon travail.

Je remercie également mes encadrants en commençant par Michel Dhome, mon directeur de thèse et Fabien Dekeyser, mon correspondant au Commissariat à l'Énergie Atomique (CEA). Il m'a été bien agréable de travailler avec eux pendant ces trois années. Je tiens à remercier sincèrement Maxime Lhuillier pour son encadrement sans faille de tous les instants et suis heureux d'avoir su le divertir.

Merci également à Patrick Sayd et Jean-Marc Lavest qui, forts de leur expérience, ont joué un rôle important dans le déroulement de ma thèse.

Je suis aussi très reconnaissant envers Éric Royer qui m'a beaucoup aidé au début de ma thèse et François Marmoiton qui a largement facilité les manipulations sur *Cycab*.

Bien sûr, je n'oublie pas Paul Checchin car il m'a fait découvrir la vision par ordinateur, et c'est grâce à lui que j'ai eu l'opportunité de faire une thèse dans ce domaine.

Je remercie les dirigeants et tous les membres permanents, temporaires, doctorants et stagiaires de l'équipe ComSee du LASMEA (Laboratoire des Sciences et Matériaux pour l'Électronique et d'Automatique) qui m'a accueilli à Clermont-Ferrand pendant la première année et demi, et du LSVE (Laboratoire des Systèmes de Vision Embarqués) du CEA LIST à Saclay où j'ai travaillé pendant la deuxième moitié du temps. Les nombreuses heures de travail en leur compagnie m'ont offert de bons moments, et je pense plus particulièrement à ceux qui ont partagé mon bureau : Lucie, Steve, Gaëtan, Laetitia, Julien, Nicolas, Pierre, Florent, Alexandre, Romain mais également Laetitia et Hala, et bien sûr tous les autres.

Un grand merci à ma famille et mes amis qui m'ont soutenu et encouragé.

Finalement, mes plus tendres remerciements s'adressent à Florence pour m'avoir supporté dans tous les sens du terme. Je ne la remercierai jamais

assez pour avoir toujours été présente à mes côtés, dans les moments les plus difficiles comme dans les meilleurs.

Résumé

Le problème de la reconstruction 3D à partir d'une séquence d'images acquise par une caméra en mouvement est un sujet important dans le domaine de la vision par ordinateur. Ce travail de thèse présente une méthode qui permet d'estimer conjointement des points 3D de la scène filmée et le mouvement de la caméra en combinant la précision des méthodes "hors-ligne" (basées sur une optimisation globale de tous les paramètres par ajustement de faisceaux) et la vitesse de calcul des méthodes incrémentales. La nouvelle approche est considérée comme une accélération des techniques classiques de reconstruction 3D qui utilisent l'ajustement de faisceaux, permettant ainsi de traiter de longues séquences vidéos.

L'algorithme développé peut être résumé de la façon suivante : détection de points d'intérêt dans les images, mise en correspondance de ces points et sous-échantillonnage temporel de la vidéo. En effet, seul un sous-ensemble d'images dites "images clef" est sélectionné pour la reconstruction des points 3D alors que la localisation de la caméra est calculée pour chaque image. Le point clef de l'approche est l'ajustement de faisceaux local : les paramètres de la reconstruction sont affinés sur la fin de la séquence uniquement, à chaque fois qu'une image est choisie comme nouvelle image clef. La méthode, initialement prévue pour les caméras perspectives, a ensuite été généralisée de manière à rendre possible l'utilisation d'autres types de caméras, comme les caméras catadioptriques ou encore les paires rigides de caméras.

Les résultats obtenus montrent que la précision atteinte est du même ordre que celle des méthodes par optimisation globale, avec des temps de calcul très réduits, ce qui permet de viser des applications d'odométrie visuelle temps-réel pour la robotique mobile ou l'aide à la conduite en automobile.

Mots-clef : reconstruction 3D, localisation, ajustement de faisceaux, temps-réel

Abstract

The Structure from Motion problem is an intense research topic in computer vision and has been the subject of much investigation. This thesis presents a method for estimating the motion of a calibrated camera and the threedimensional geometry of the filmed environment. The main idea is to take advantage of both offline methods (based on an optimization of all 3D parameters by global bundle adjustment) and fast incremental methods. The new approach may be seen as an acceleration of conventional 3D reconstruction techniques that make use of bundle adjustment, and thus enables to treat very long video sequences.

The introduced algorithm may be summarized as follows : interest points detection and matching between frames, sub-sampling of the video into "key frames", full 3D reconstruction of these key frames (3D points and camera poses), and localization of all frames. The keystone of the method is the local bundle adjustment : reconstruction parameters are refined at the end of the sequence only, for all current frame selected as key frame. This method is applied initially to a perspective camera model, then extended to a generic camera model to describe most existing kinds of cameras like catadioptric cameras or stereo rigs.

Experiments have shown that results are very similar to those obtained by methods with global optimisation, with much lower computing times. We can envisage applications like real-time visual odometry for mobile robots or car assisted driving.

Key-words : Structure from Motion, localization, bundle adjustment, real-time

Table des matières

Table des figures

Liste des tableaux

Introduction

La robotique est un domaine scientifique et technologique dans lequel on cherche souvent à reproduire le fonctionnement humain de manière artificielle, à partir de capteurs, de calculateurs et d'actionneurs. Elle est très liée à la vision par ordinateur qui est le domaine précis où l'on analyse les images produites par le capteur caméra pour en extraire des informations importantes. On cherche alors à imiter le système naturel constitué de l'oeil (capteur) et du cerveau (analyse). On arrive déjà à faire mieux dans certaines applications : en métrologie par exemple (domaine quantitatif) où l'on sait mesurer des distances inférieures au millimètre alors qu'on est incapable de le faire à l'oeil nu. Pour d'autres applications (domaine qualitatif), de grands progrès sont encore à réaliser : par exemple en reconnaissance et détection d'objets.

Avec la montée en puissance des ordinateurs et la démocratisation des capteurs numériques, la vision par ordinateur ne pouvait que connaître un très grand développement. Il y a une vingtaine d'années, il était impossible de charger une image entière d'un million de pixels dans la mémoire vive d'un ordinateur. Aujourd'hui, même les téléphones mobiles sont équipés d'appareils photos intégrés et sont capables de stocker et de traiter des dizaines d'images. Il s'est vendu un milliard de téléphones portables dans le monde en 2006, dont la moitié étaient équipés d'un appareil photo intégré. De nombreux autres produits intègrent aujourd'hui des caméras numériques (automobiles, ordinateurs de poche, baladeurs, etc ...) et cela montre le gigantesque potentiel de diffusion des algorithmes développés dans ce domaine.

Ce travail de thèse concerne particulièrement la reconstruction 3D à partir d'une séquence d'images. C'est un problème majeur en vision par ordinateur qui consiste à estimer la géométrie tridimensionnelle de la scène filmée ainsi que la position et l'orientation du système de prise de vue. On peut voir ce problème comme un problème de localisation du capteur et de cartographie de l'environnement. Ces aspects de localisation et cartographie sont bien connus à ce jour et plébiscités par le grand public. Pour s'en convaincre, il

suffit de remarquer le succès des logiciels *Géoportail*[1] et *Google earth*[2] avec lesquels chacun peut gratuitement visualiser ses trajets et observer n'importe quel lieu sur la planète à partir d'une collection d'images aériennes ou satellitaires. Les systèmes de géo-positionnement et d'aide à la navigation ont récemment connu un très grand succès, notamment les récepteurs GPS[3]. En France, il s'en est vendu plus de 1,23 million en 2006. Le marché a véritablement explosé avec une augmentation de 290% par rapport à l'année précédente, ce qui prouve le véritable engouement du grand public pour ces systèmes de localisation.

Localisation par vision

Il y a deux manières de concevoir le problème de localisation à partir d'une caméra. La première consiste à considérer la localisation comme un problème global, à la manière d'un positionnement par GPS. Par exemple, si l'image qui provient du capteur est une prise de vue d'un monument, alors il est possible de reconnaître automatiquement de quel monument il s'agit (à partir d'une base de données), et de déterminer ainsi où l'on se trouve. Cela nécessite bien sûr de connaître tous les lieux où l'on peut se déplacer. La deuxième approche est plus locale. Elle consiste non pas à trouver une position absolue mais plutôt une position relative par rapport à une localisation connue. Par exemple, considérons qu'on s'est déplacé d'un point A connu vers un point B inconnu et qu'il n'est pas possible de déterminer la position en B par la méthode globale. On peut tout de même estimer la position B si l'on est capable d'estimer le mouvement réalisé par la caméra pour aller de A à B. C'est cette deuxième approche qui est exploitée dans ces travaux de thèse en calculant une reconstruction 3D.

Localisation et reconstruction 3D

Le principe est le suivant : on considère une caméra qui se déplace dans un environnement inconnu et qui acquiert des images de cet environnement.

[1] *Géoportail* est un portail web permettant d'accéder à des données publiques localisées, sur l'ensemble du territoire français, dans une interface cartographique. `http ://www.geoportail.fr/`

[2] *Google Earth* est un logiciel de la société Google permettant une visualisation de la terre avec un assemblage de photographies aériennes ou satellitaires. `http ://earth.google.com/`

[3] GPS : Global Positioning System, est le principal système de positionnement par satellite mondial

A partir des images, le but est de calculer conjointement la trajectoire de la caméra et la structure 3D de la scène filmée. En effet, les deux calculs sont très liés et si l'on dispose de l'une ou l'autre des informations, la deuxième est facilement calculable. De nombreux algorithmes ont été développés mais ils ne peuvent pas traiter des longues séquences vidéos pour deux principales raisons :

- le manque de précision dû à l'accumulation des erreurs
  ou
- l'impossibilité de faire les calculs dans des temps raisonnables.

Contribution

Au cours de ces travaux, on s'est intéressé à développer une méthode de reconstruction 3D et localisation qui limite l'accumulation des erreurs mais qui soit réalisable en temps-réel. Pour cela, on a cherché à adapter une technique classique (l'ajustement de faisceaux) très précise et habituellement réalisée dans des applications hors-ligne. La complexité de calcul est telle qu'il est impossible d'utiliser sa forme standard en temps-réel sur des séquences vidéo de plusieurs dizaines d'images. On a voulu montrer que sous sa forme locale couplée à un algorithme de reconstruction 3D incrémental, l'ajustement de faisceaux conserve ses propriétés de précision avec une complexité réduite. On s'est attaché à valider les performances en comparant nos résultats avec ceux obtenus par une méthode de vision plus classique ou avec les données provenant d'un récepteur GPS différentiel.

Dans un deuxième temps, on a cherché à généraliser le principe de reconstruction 3D précise en temps-réel et à vérifier que la méthode développée à l'origine pour les caméras perspectives était adaptable à d'autres types de caméras. Le premier intérêt de cette généralisation était de pouvoir utiliser différents types de caméra sans changer de méthode de reconstruction. Le second était de considérer un ensemble rigide de plusieurs caméras comme une seule et même caméra à plusieurs centres de projection.

Structure du document

Ce mémoire de thèse s'articule autour de quatre chapitres :

1. Un rappel sur les principes et les méthodes couramment utilisés en vision par ordinateur pour l'obtention des informations géométriques 3D.
2. Une revue des grandes familles d'algorithmes existants.

3. L'explication détaillée de la méthode complète de reconstruction 3D développée pour une caméra perspective en mouvement.
4. La généralisation de cette méthode aux caméras génériques.

Chapitre 1

Principes et méthodes de base

Dans ce premier chapitre, on définit le modèle de caméra utilisé et on présente les outils mathématiques couramment employés en reconstruction 3D pour l'obtention des modèles géométriques. On aborde les problèmes élémentaires comme le calcul de pose ou la triangulation de points mais également les problèmes plus complexes comme le calcul de solutions optimales en présence de bruit ou encore l'ajustement de faisceaux.

1.1 La caméra perspective

1.1.1 Le modèle sténopé

Les capteurs de vision fréquemment utilisés pour l'acquisition de séquences vidéos peuvent généralement être décrits par le modèle "sténopé" encore appelé "trou d'épingle" ou "perspectif". Dès l'antiquité, on utilisait le principe du sténopé pour observer les éclipses solaires dans une chambre noire. Tous les rayons de lumière passent par un seul et même point : le centre optique ou centre de projection. Comme illustré sur la Figure 1.1, l'image se forme sur le plan image (mur ou plaque photosensible) naturellement situé derrière le centre optique à une distance f de celui-ci. L'image obtenue sur ce plan est alors inversée par rapport à la scène.
Pour corriger ce problème, on place artificiellement le plan image devant le centre optique.

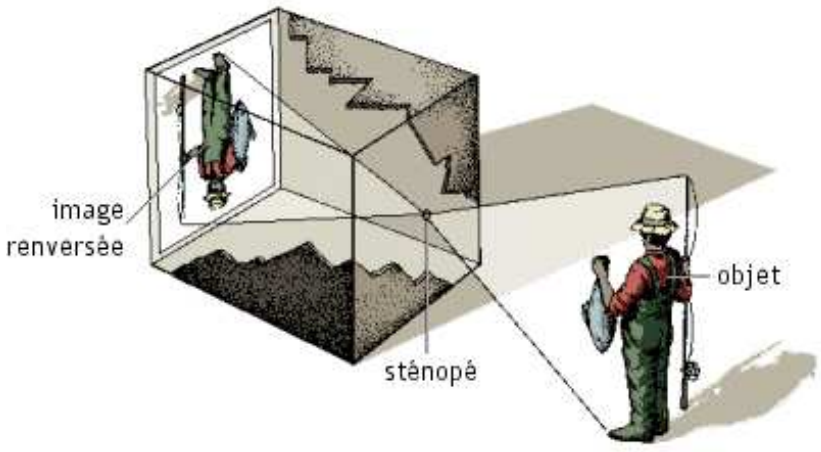


FIG. 1.1 – Principe de la chambre noire. Source : Encyclopédie Microsoft Encarta.

1.1.2 Les paramètres intrinsèques

Une caméra décrite par ce modèle est alors caractérisée par ses paramètres internes qui sont (voir Figure 1.2) :

- la distance focale f (distance orthogonale du centre optique au plan image).
- le point principal $\mathbf{u} = (u_0, v_0)$ qui est l'intersection de l'axe optique et du plan image (supposés orthogonaux).
- le rapport $\kappa = dy/dx$ des dimensions verticale et horizontale des pixels du capteur photosensible.

Si on pose $f_x = f/d_x$, $f_y = f/d_y$ et si d_x est connu (fourni par le constructeur de la caméra ou fixé arbitrairement à 1), alors le couple de paramètres (f, κ) peut être remplacé par le couple (f_x, f_y) et les paramètres intrinsèques se ramènent à (f_x, f_y, u_0, v_0).

1.1.3 Les paramètres extrinsèques

Dans ce paragraphe, il convient de définir deux différents repères tridimensionnels : le repère du monde noté R_w qui est le repère global contenant la scène, et le repère lié à la caméra noté R_c. Le changement de repère $R_w \rightarrow R_c$ correspond à la position ou "pose" de la caméra dans le repère du monde au moment de la prise de vue. Cette pose est modélisée par les paramètres extrinsèques de la caméra. Ils peuvent être représentés par :

- un vecteur de translation $\mathbf{t}$ de taille 3×1 indiquant la position du centre

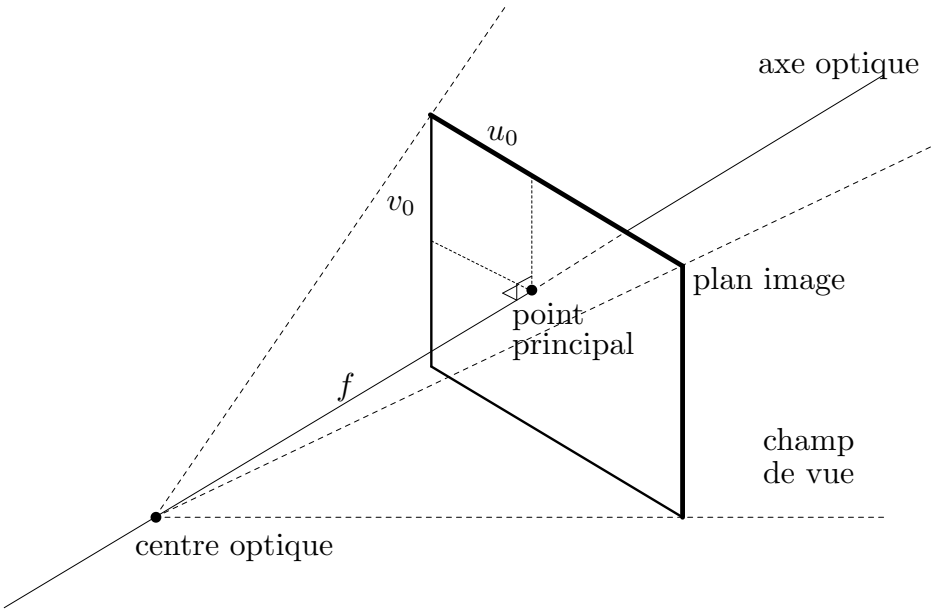


FIG. 1.2 – Le modèle sténopé, paramètres intrinsèques.

optique de la caméra dans le repère du monde.

– une matrice de rotation $\mathtt{R}$ de taille 3×3 indiquant l'orientation de la caméra dans le même repère. Elle peut être déterminée à partir des angles d'EULER : α, β et γ.

$$\mathtt{R} = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}.$$

On notera $C = (\mathtt{R}, \mathbf{t})$ la pose de la caméra.

1.1.4 Changement de repère monde/caméra

Un vecteur $\mathbf{V}_c$ dans le repère caméra s'écrit $\mathbf{V}_w = \mathtt{R}\mathbf{V}_c + \mathbf{t}$ dans le repère monde et à l'inverse, un vecteur $\mathbf{V}_w$ dans le repère monde s'écrit $\mathbf{V}_c = \mathtt{R}^\top (\mathbf{V}_w - \mathbf{t})$ dans le repère caméra.

Remarque : Les coordonnées homogènes permettent de linéariser les équations de changement de repère ainsi que les équations de projection. Soit $\mathbf{V}$ un vecteur de l'espace, de coordonnées cartésiennes $\mathbf{V} = [x\ y\ z]^\top$. On appelle coordonnées homogènes de $\mathbf{V}$ tout quadruplet de réels $\bar{\mathbf{V}} = [X\ Y\ Z\ T]^\top$ tels que : $x = X/T$, $y = Y/T$, et $z = Z/T$. Les coordonnées homogènes ne sont pas uniques, et sont définies à une constante multiplicative près. De

la même manière, on définit les coordonnées homogènes pour un vecteur du plan comme un triplet de réels. En coordonnées homogènes, le changement de repère s'écrit :

$$\begin{aligned} &\bullet\ \bar{\mathbf{V}}_w \equiv \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathtt{0}_{1\times 3} & 1 \end{bmatrix} \bar{\mathbf{V}}_c \\ &\bullet\ \bar{\mathbf{V}}_c \equiv \begin{bmatrix} \mathtt{R}^\top & -\mathtt{R}^\top\mathbf{t} \\ \mathtt{0}_{1\times 3} & 1 \end{bmatrix} \bar{\mathbf{V}}_w \end{aligned} \tag{1.1}$$

où $\mathtt{0}_{1\times 3}$ est la matrice nulle de taille 1×3 et le signe $\equiv$ désigne l'égalité à un facteur multiplicatif près.

1.1.5 La projection perspective d'un point 3D

L'image fournie par une caméra est formée à partir de l'ensemble des projections sur le plan image des points 3D de la scène situés dans le champ de vue (voir Figure 1.3). Pour chaque point 3D visible $\mathbf{P}$, un rayon lumineux dirigé vers le centre optique frappe le plan image en un point $\mathbf{p} = [X\ Y\ Z]^\top$ (coordonnées homogènes). $x = X/Z$ et $y = Y/Z$ sont les coordonnées de $\mathbf{p}$ en pixels dans l'image. On peut décomposer la projection de $\mathbf{P}$ en trois étapes :

1. le passage du repère monde au repère caméra

$$\bar{\mathbf{P}}_c \equiv \begin{bmatrix} \mathtt{R}^\top & -\mathtt{R}^\top\mathbf{t} \\ \mathtt{0}_{1\times 3} & 1 \end{bmatrix} \bar{\mathbf{P}}_w \tag{1.2}$$

   où $\bar{\mathbf{P}}_c$ et $\bar{\mathbf{P}}_w$ sont respectivement les coordonnées homogènes de $\mathbf{P}$ dans le repère caméra et le repère monde.

2. la projection 3D/2D du point dans le même repère

$$\bar{\mathbf{p}}_c \equiv [\mathtt{I}_{3\times 3}|\mathtt{0}_{3\times 1}]\, \bar{\mathbf{P}}_c \tag{1.3}$$

3. le passage en coordonnées pixels

$$\bar{\mathbf{p}} \equiv \mathtt{K}\bar{\mathbf{p}}_c \tag{1.4}$$

   où $\mathtt{K} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$ est la matrice des paramètres intrinsèques de la caméra.

Globalement, la projection du point $\mathbf{P}$ s'obtient par

$$\bar{\mathbf{p}} \equiv \mathtt{K}\mathtt{R}^{\top} \left[\mathtt{I}_3 | -\mathbf{t}\right] \bar{\mathbf{P}}_w \tag{1.5}$$

ou plus simplement

$$\bar{\mathbf{p}} \equiv \mathtt{C}\bar{\mathbf{P}}_w \tag{1.6}$$

où $\mathtt{C}$ est la matrice de projection de taille 3×4 associée à la caméra.
Les coordonnées x, y en pixels sont données par l'équation de projection :

$$[x\ y]^{\top} = \pi\left(\mathtt{C}\bar{\mathbf{P}}_w\right) = \pi\left(\mathtt{K}\mathtt{R}^{\top}\left[\mathtt{I}_3 | -\mathbf{t}\right]\bar{\mathbf{P}}_w\right) \tag{1.7}$$

avec π fonction $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ telle que $\pi\left([X\ Y\ Z]^{\top}\right) = [X/Z\ Y/Z]^{\top}$.

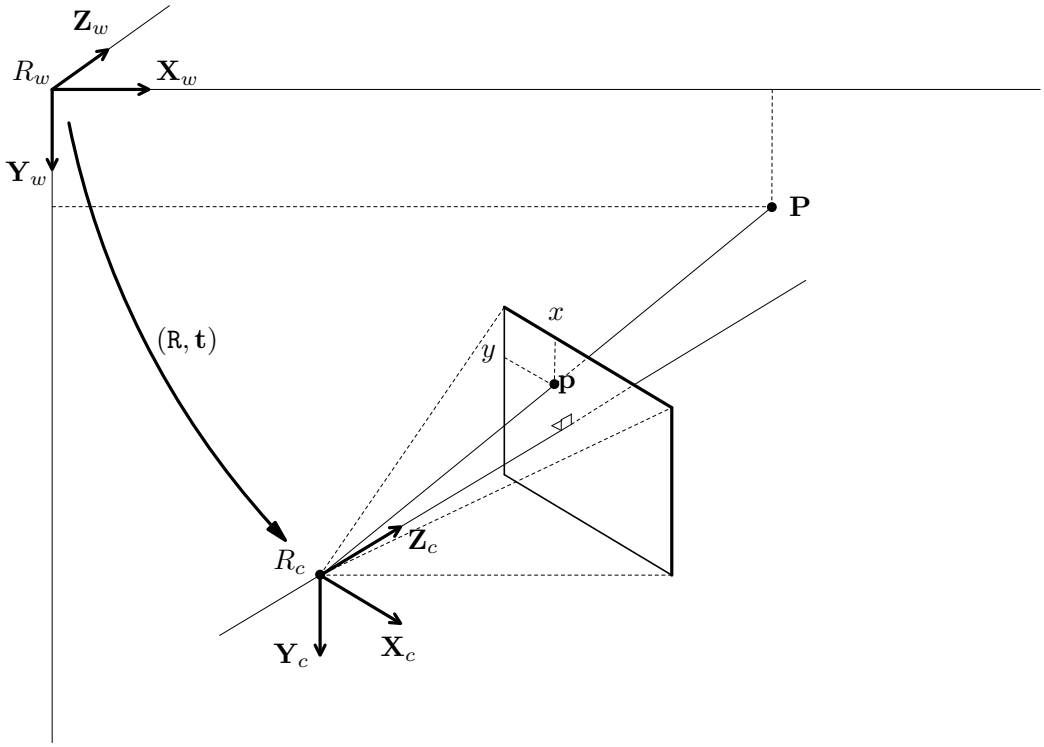


FIG. 1.3 – Systèmes de coordonnées utilisés et projection d'un point 3D.

1.1.6 Paramètres de distorsion

Le modèle de projection perspective correspond à une approximation du modèle optique des lentilles minces. Cependant, les systèmes de vision actuels ne correspondent pas exactement à ce modèle, notamment si l'objectif utilisé

est un objectif grand angle à courte focale. Dans ce cas, l'optique introduit des distorsions dans l'image et le modèle sténopé atteint ses limites. La distorsion se manifeste visuellement par des déformations dans l'image, et notamment les droites qui deviennent courbes. Pour pouvoir modéliser correctement la projection, il est indispensable de tenir compte de ces distorsions. Pour une caméra réelle, le rayon lumineux qui vient frapper le capteur au point $\mathbf{p}$ diffère du point $\mathbf{p}_{sd}$ qui serait excité par la même caméra sans l'effet de distorsion (sténopé parfait).

Modéliser la distorsion revient à déterminer la fonction qui à $\mathbf{p} = (x, y)$ associe $\mathbf{p}_{sd} = (x_{sd}, y_{sd})$ (corrigé de la distorsion). La distorsion a une composante tangentielle et une composante radiale par rapport au point principal $\mathbf{u}$. Afin de simplifier le modèle, on considère que la distorsion tangentielle est négligeable et on s'intéresse uniquement à la composante radiale de la distorsion. Celle-ci peut être modélisée par un polynôme de degré 5 dont les coefficients sont a_1 à a_5 (paramètres de distorsion). Si on note r la distance normalisée au carré entre $\mathbf{p}$ et $\mathbf{u}$ telle que $r = \left\| \dfrac{\mathbf{p}-\mathbf{u}}{f_x} \right\|^2 = \left(\dfrac{x-u_0}{f_x} \right)^2 + \left(\dfrac{y-v_0}{f_x} \right)^2$, alors le point corrigé est

$$\mathbf{p}_{sd} = \mathbf{p} + (a_5.r^5 + a_4.r^4 + a_3.r^3 + a_2.r^2 + a_1.r)(\mathbf{p} - \mathbf{u})$$

et ses coordonnées sont obtenues par

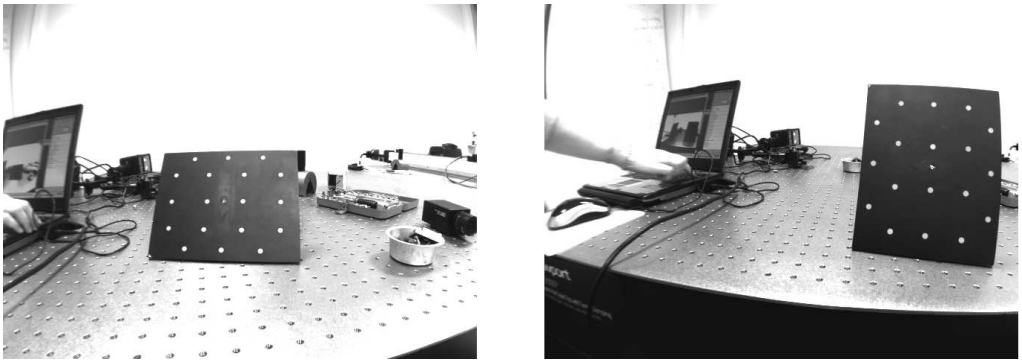
$$\begin{bmatrix} x_{sd} \\ y_{cd} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + (a_5.r^5 + a_4.r^4 + a_3.r^3 + a_2.r^2 + a_1.r) \begin{bmatrix} x-u_0 \\ y-v_0 \end{bmatrix}. \tag{1.8}$$

Remarque : Pour donner un ordre de grandeur, les caméras utilisées dans ces travaux ont des champs de vue particulièrement larges ($> 100°$). Dans notre cas, la distorsion peut atteindre plusieurs dizaines de pixels. Les paramètres de distorsion a_1 à a_5 sont déterminés lors du calibrage de la caméra présenté au prochain paragraphe.

1.1.7 Étalonnage des caméras

L'étalonnage (ou calibrage) des caméras consiste à déterminer de la façon la plus précise possible les paramètres intervenant dans l'expression analytique de l'équation de projection 1.7. Il s'agit des quatre paramètres intrinsèques (f_x, f_y, u_0, v_0) et des six paramètres extrinsèques (3 pour $\mathtt{R}$ + 3 pour $\mathbf{t}$), soit dix paramètres. Il faut également ajouter les cinq paramètres de distorsion radiale, ce qui fait un total de 15 paramètres. Dans les travaux présentés

dans ce rapport, on considère que la caméra utilisée est calibrée, c'est-à-dire que les paramètres (f_x, f_y, u_0, v_0) ainsi que les paramètres de distorsion $(a_1, a_2, a_3, a_4, a_5)$ sont parfaitement connus. En fait, ceux-ci sont déterminés lors d'une étape de calibrage faite au préalable. Le calibrage est effectué en prenant plusieurs images d'une mire et en cherchant les paramètres de la caméra qui minimisent l'erreur de reprojection de chaque point de la mire dans chaque image. La méthode que nous avons employée est une méthode d'autocalibrage par ajustement de faisceaux utilisant une mire plane selon le procédé donné par Lavest *et al.* [38]. Deux images utilisées pour le calibrage sont présentées sur la Figure 1.4(a) et les résultats obtenus dans le Tableau 1.4(b) de la Figure 1.4. La mire est une plaque noire sur laquelle des pastilles rondes photo-réfléchissantes ont été disposées et l'approche se base sur une détection de grande précision des points de la mire. Une dizaine d'images sont utilisées pour le calibrage en essayant de parcourir au maximum toute la surface de l'image.



(a) 2 exemples d'images de la mire plane utilisée pour le calibrage.

Caméra	AVT Guppy F-033B
Objectif	Tamron 13FA28T
Image	640×480
Nb d'images	9
Nb de mesure	128

	valeur	σ
f_x (pix)	383.67	4.99e-01
f_y (pix)	383.45	5.52e-01
u_0 (pix)	354.64	3.11e-01
v_0 (pix)	273.40	4.56e-01
a_1	+3.451e-01	2.927e-03
a_2	-1.787e-01	1.499e-02
a_3	+5.260e-01	3.562e-02
a_4	-4.179e-01	3.755e-02
a_5	+1.746e-01	1.441e-02

(b) Résultats du calibrage sur les paramètres intrinsèques et les paramètres de distorsion.

Fig. 1.4 – Étalonnage d'une caméra : (a) images de la mire plane. (b) résultats.

1.2 Le problème de la reconstruction 3D

1.2.1 Principe

La reconstruction 3D consiste à retrouver l'information tridimensionnelle des objets perdue lors de l'acquisition d'images, c'est-à-dire la projection de la scène sur un plan. En effet, une caméra se déplaçant dans un monde 3D collecte un ensemble d'images qui sont des données bidimensionnelles. Pourtant, à la fois la trajectoire de la caméra et le monde dans lequel elle évolue sont en 3D. Le processus de reconstruction consiste souvent à déterminer conjointement la structure de la scène et la position des caméras, uniquement à partir des images (Figure 1.5).
Selon le contexte, trois types de reconstruction peuvent être considérés, sans lien avec le modèle de caméra utilisé. La notion de reconstruction "projective" est utilisée lorsque les coordonnées des objets sont définies dans un repère projectif. La reconstruction est alors définie à une transformation projective près et ne contient pas d'information d'angle, de longueur, ou de parallélisme. De même, une reconstruction "affine"est définie à une transformation affine près et ne contient que des informations affines (rapport de longueurs et parallélisme). Enfin, le dernier type de reconstruction et celui qui nous intéresse particulièrement est la reconstruction "euclidienne". Tous les éléments 3D sont référencés dans un repère euclidien, qui nous est plus familier et où les notions d'angle et de distance sont présentes.
Dans le cas idéal, une reconstruction serait totale et permettrait de modéliser entièrement l'environnement 3D. Pour cela, il faudrait observer l'ensemble des points de l'espace. En effet, il est bien sûr possible de reconstruire uniquement les objets observés par la caméra, c'est-à-dire situés dans le champ de vue et sans subir d'occultation par d'autres objets. Une telle reconstruction totale est difficilement concevable. Par exemple, notre cerveau sait que lorsqu'on observe une maison, on se trouve devant un objet volumique en 3 dimensions. L'ordinateur équipé d'une caméra, lui ne verra par exemple qu'une façade complètement plane.
Même s'il est envisageable d'utiliser l'information d'intensité lumineuse d'une seule image pour déterminer la forme des objets (on parle alors de "Shape from Shading" [106]), la vision tridimensionnelle n'est souvent possible que si l'on a deux points de vue différents d'une même scène. C'est probablement ce que fait le cerveau humain en combinant les informations provenant des deux yeux. On peut alors employer plusieurs caméras (stéréo vision, vision trinoculaire, etc) ou bien utiliser le mouvement d'une seule et même caméra. Dans

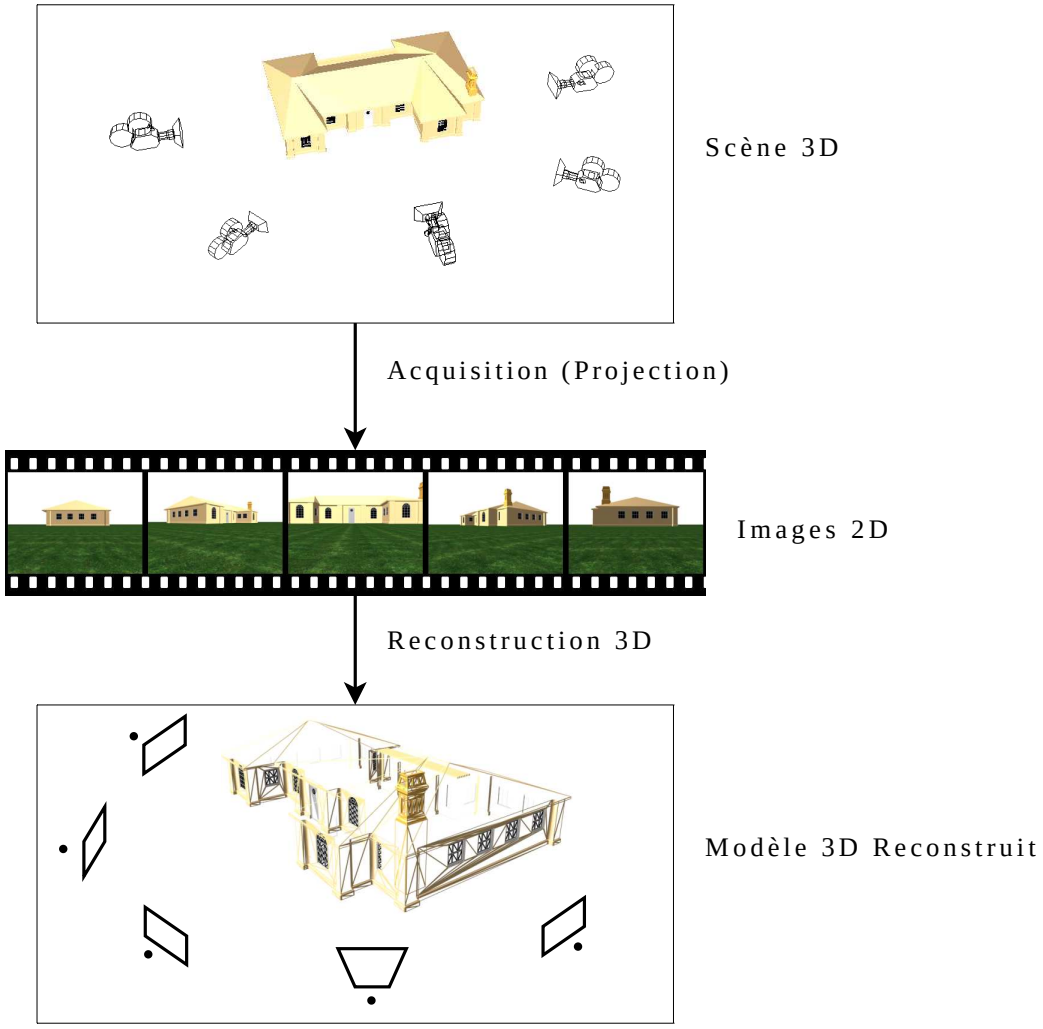


FIG. 1.5 – Principe de la reconstruction 3D à partir d'une séquence vidéo.

ce cas, on parle de "structure à partir du mouvement" ou plus communément de SfM pour "Structure from Motion" en anglais. Pour un objet détecté dans une image, il faut alors être capable de dire si cet objet est visible dans une autre image, et quelle est sa nouvelle position dans cette image. Encore une fois, cette tâche est évidente pour le cerveau, mais très compliquée à réaliser avec un ordinateur. En effet, il nous est immédiat de voir si un objet vu par l'oeil gauche est le même qu'un objet vu par l'oeil droit. En vision par ordinateur, on parle de "recalage" ou "mise en correspondance" d'images et c'est un problème fondamental, objet de nombreux travaux. C'est l'étape de

traitement d'image initiale pour la plupart des algorithmes de reconstruction. Une fois la mise en correspondance réalisée, on dispose de disparités 2D (positions différentes des mêmes objets dans les images) qui vont permettre d'accéder simultanément ou indépendamment au calcul du déplacement de la caméra (ego-motion estimation ou pose relative) et au calcul de la position 3D des objets (triangulation) - les deux calculs étant souvent liés.
Dans cette section nous allons définir la notion de primitive et nous intéresser particulièrement à la primitive la plus simple qui est le point. Nous allons voir comment il est possible de détecter des points particuliers et les mettre en correspondances entre plusieurs images. Ensuite, nous pourrons formuler le problème de la reconstruction 3D d'un nuage de points à partir d'une séquence d'images.

1.2.2 Les primitives

Si l'on dispose de deux images d'une même scène, on peut essayer de faire correspondre chaque pixel (ou presque) d'une image à un pixel de l'autre image et de reconstruire l'ensemble des pixels. On parle de mise en correspondance et reconstruction dense (par exemple [68, 86, 72], ...) ou quasi dense [45]. Au contraire, on peut seulement s'intéresser à quelques objets intéressants de l'image. Ces objets intéressants sont souvent appelés "amers" ou "primitives" et ils représentent les structures géométriques élémentaires du modèle. Ils peuvent être de simples points répartis dans l'espace de manière complètement libre ou alors des éléments plus structurés. De nombreux auteurs se sont intéressés aux cas particuliers des droites dans l'espace [15, 88, 28, 55, 3] ou aux structures planaires [1, 2]. En milieu urbain par exemple, l'estimation du mouvement d'une caméra embarquée sur un véhicule peut être réalisée grâce à la détection du plan principal de la route [93]. Pour la reconstruction d'objets industriels manufacturés ou d'objets d'art, il est possible de se baser sur une modélisation qui utilise des courbes paramétriques de type spline ou NURBS (Spline de base non uniforme rationnelle) [56]. Dans les applications de reconstruction 3D de bâtiments architecturaux, il est intéressant d'utiliser les segments, les coins ou les plans qui sont très présents. Dans ce domaine, CORNOU *et al.* [11] ont étudié de manière générale l'ajout de contraintes géométriques dans la reconstruction 3D. Cependant, la détection des primitives complexes peut s'avérer compliquée et échouer si la forme du modèle ne correspond pas vraiment à la réalité (angles droits non respectés par exemple). De même, des contraintes trop fortes peuvent entraîner un manque de précision. L'utilisation de points libres offre un plus

grand nombre de degrés de liberté et peut se faire quelle que soit la géométrie de la scène, à condition toutefois qu'il y ait suffisamment d'information de texture dans les images.

1.2.3 Détection et mise en correspondance de points d'intérêt

Au cours de ce travail de thèse, nous nous sommes intéressés uniquement à la reconstruction de points libres. Dans ce cas, la mise en correspondance d'images consiste à apparier des points entre deux images. Cette étape se décompose souvent en 3 parties :

- La détection des points d'intérêt dans chaque image. C'est une étape primordiale car ce sont les points détectés qui sont susceptibles d'être reconstruits en 3D, à condition de trouver leurs points homologues dans une autre image.
- Le calcul des descripteurs locaux. Ils permettent de résumer l'information contenu dans le voisinage d'un point et ainsi mesurer facilement la ressemblance avec un autre point.
- L'appariement de points, qui consiste à créer des paires de points homologues (un point dans chaque image). Si l'appariement est correct, alors les deux points de la paire correspondent bien au même point 3D de la scène.

Détection de points d'intérêt

Les points dits "d'intérêt" ont été étudiés dès la fin des années 1970 par Beaudet [6] puis Moravec [62]. Dès lors, on s'est basé sur la détection des points d'intérêt pour les calculs du flot optique [16] ou les calculs de géométrie épipolaire [107]. L'utilisation de tels points est désormais très courante en traitement d'image et en vision par ordinateur et elle est essentielle à de nombreuses applications : détection de mouvement, suivi, modélisation 3D, reconnaissance d'objets, etc. Un point d'intérêt est un point qui a une position bien définie dans une image et qui correspond à un point de l'espace détectable de manière robuste au changement de point de vue ou d'éclairage. Il en existe plusieurs types. Ils peuvent être simplement des points de l'image où l'intensité lumineuse $I(x, y)$ est localement minimale ou maximale. Il peuvent être également des points de contour où la courbure est maximale (approches basées contour). Les plus courants sont les "coins", où le gradient de l'intensité lumineuse $I(x, y)$ est fort dans deux directions. Ces points sont

facilement identifiables dans les images car ils correspondent souvent aux coins des objets de la scène ou aux détails de texture. La qualité d'un détecteur se mesure à sa répétabilité, c'est-à-dire son aptitude à détecter le même point de l'espace dans plusieurs images prises dans des conditions différentes. Une étude comparative de plusieurs détecteurs [87] a montré qu'un détecteur basé sur celui de HARRIS et STEPHENS [25] offrait la meilleure répétabilité.

Alors que le détecteur de MORAVEC [62] utilise la somme des distance au carré entre une imagette centrée autour d'un point et une imagette légèrement décalée, le détecteur de HARRIS prend en compte les courbures principales de la fonction d'autocorrélation du signal. La matrice de HARRIS (notée $\mathtt{A}$) est calculée pour chaque pixel (x, y) à partir des dérivées premières de $I(x, y)$.

$$\mathtt{A}(x,y) = \begin{bmatrix} \left\langle \frac{\partial I}{\partial x}(x,y) \right\rangle^2 & \left\langle \frac{\partial I}{\partial x}(x,y) \right\rangle \left\langle \frac{\partial I}{\partial y}(x,y) \right\rangle \\ \left\langle \frac{\partial I}{\partial x}(x,y) \right\rangle \left\langle \frac{\partial I}{\partial y}(x,y) \right\rangle & \left\langle \frac{\partial I}{\partial y}(x,y) \right\rangle^2 \end{bmatrix}$$

où l'opérateur $\langle\ \rangle$ représente le lissage de l'image en x et y par une gaussienne de variance σ^2. Cela permet d'améliorer le calcul des dérivées qui est très sensible au bruit, et plusieurs techniques de lissages sont possibles. Le critère de HARRIS et STEPHENS est donné par :

$$C(x,y) = det(\mathtt{A}) - k.trace^2(\mathtt{A}) \tag{1.9}$$

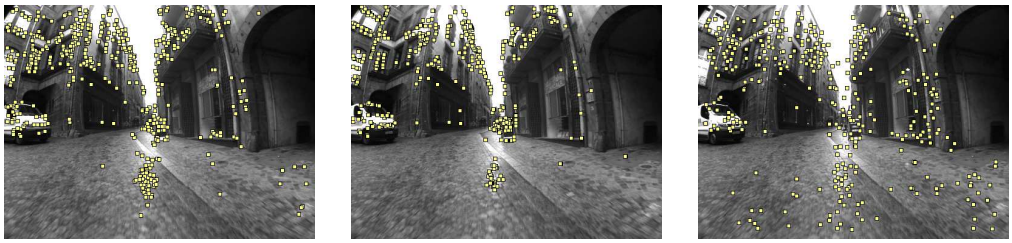
où k est un coefficient supplémentaire bien choisi (k est pris de l'ordre de 0.04 selon HARRIS). Le détecteur développé par SHI et TOMASI [90] est une variante du détecteur de HARRIS qui utilise le fait que lorsque les valeurs propres réelles de $\mathtt{A}$ (notées λ_1 et λ_2) sont grandes, alors la probabilité d'avoir détecté un point d'intérêt est forte. Le critère est alors défini de la manière suivante :

$$C(x,y) = min\left(\lambda_1(x,y), \lambda_2(x,y)\right). \tag{1.10}$$

D'après les auteurs, ce détecteur est meilleur que celui de HARRIS si les images subissent une transformation affine. Dans ce sens, des détecteurs invariants aux transformations affines et aux changements d'échelle ont été étudiés. Par exemple, MIKOLAJCZYK et SCHMID [59] ont introduit un nouveau détecteur qui utilise une approche multi-échelle et permet de caractériser le voisinage du point d'intérêt pour le calcul d'un descripteur local. La même idée est exploitée par LOWE [51] pour détecter des points en leur associant une échelle et une orientation. Ces détecteurs sont très performants en

termes de qualité de détection mais ils sont beaucoup plus coûteux en termes de temps de calcul.

Plus récemment, une méthode nommée FAST (pour Features from Accelerated Segment Test) [79] a vu le jour. Le détecteur considère les pixels sur un cercle de Bresenham de rayon r autour d'un point candidat. Un coin est choisi s'il existe un nombre n de pixels contigus sur le cercle qui sont tous plus brillants (ou tous plus sombres) que le pixel candidat. Ce détecteur est considéré comme produisant des points d'intérêt très stables avec $r = 3$ (ce qui correspond à un cercle de 16 pixels de circonférence) et $n = 9$. La méthode prévoit également d'optimiser par une méthode d'apprentissage l'ordre dans lequel les pixels sont testés. En même temps, Bay *et al.* [4] ont proposé le détecteur Fast Hessian basé sur l'approximation de la matrice Hessienne en utilisant les images intégrales.
La Figure 1.6 montre un exemple d'image pour laquelle on a appliqué trois détecteurs différents : Harris, FAST, et Fast Hessian.



(a) Harris (b) FAST (c) Fast Hessian

Fig. 1.6 – Points d'intérêt avec 3 détecteurs différents.

Les descripteurs locaux

Un descripteur local permet de résumer l'information contenue dans le voisinage d'un point d'intérêt. Il permet de reconnaître le point grâce à son environnement proche. La méthode la plus simple pour cela est de prendre pour descripteur une imagette centrée autour du point. Comparer deux points revient à comparer l'intensité lumineuse dans les deux imagettes correspondantes et calculer un score de ressemblance. Celui-ci peut être obtenu par simple somme des différences au carré d'intensité pour chaque pixel ou par un calcul de corrélation plus sophistiqué (ex : ZNCC Zero Normalized Cross Correlation). La deuxième méthode est plus robuste aux changements d'illumination mais elle n'est pas (tout comme la première) invariante aux changements de perspective. Elle est pourtant intéressante car elle demande peu

de temps de calcul.
De nombreux autres descripteurs ont été développés, basés sur des techniques très différentes les unes des autres. Une étude comparative de plusieurs approches a été menée par MIKOLAJCZYK et SCHMID [60]. Le descripteur SIFT (Scale Invariant Feature Transform) introduit par LOWE [51] est considéré dans cette étude comme le plus performant. Ce descripteur est basé sur une approche multi-échelle et sur la construction d'histogrammes de l'orientation des gradients autour du point d'intérêt. Les informations sont stockées dans un vecteur de dimension 128 (un tableau 4×4 d'histogrammes et 8 directions pour chaque point). Ainsi, comme son nom l'indique, il est invariant aux changements d'échelle mais aussi aux changements affines d'illuminations, et aux rotations. Il permet ainsi d'apparier des points avec des changements de point de vue assez importants. Dans [60], MIKOLAJCZYK et SCHMID proposent une variante de SIFT appelée GLOH (Gradient Location and Orientation Histogram) qui améliore la robustesse et la distinctivité. Pour cela, la taille du vecteur de description est augmentée à 272 puis diminué à 128 grâce à une analyse en composantes principales. Le principal inconvénient de ces méthodes est le temps de calcul relativement élevé. L'analyse en composantes principales a également été utilisée par KE et SUKTHANKAR [35] pour améliorer et accélérer SIFT en utilisant un descripteur de seulement 20 composantes. Très récemment, une méthode différente dénommée SURF (Speeded Up Robust Features) a été publié par BAY *et al.* [4]. Cette méthode propose le détecteur Fast Hessian et un descripteur qui repose sur une distribution d'ondelettes de HAAR 2D dans le voisinage des points d'intérêt.

L'appariement de points

Considérons maintenant que l'on dispose de deux images de la même scène prises selon des points de vue différents. Pour chacune d'elles, on suppose que l'on a détecté des points d'intérêt et calculé les descripteurs correspondants. La phase d'appariement proprement dite consiste à établir une correspondance entre les points ayant des caractérisations similaires entre les deux images. La similarité entre deux points $\mathbf{p}^1$ et $\mathbf{p}^2$ se mesure grâce à un score calculé à partir des deux descripteurs associés. Un score élevé (supérieur à un seuil s) peut signifier que les deux points sont des points homologues qui correspondent à la projection du même point $\mathbf{P}$ de la scène. Pour deux images, le but est donc de créer une liste contenant l'ensemble des points homologues. Il existe plusieurs stratégies pour cela, et pas de méthode universelle. Pour un point $\mathbf{p}^1$ de l'image 1, on peut rechercher son correspondant $\mathbf{p}^2$ dans toute

l'image 2 ou alors se contenter d'une zone de recherche déterminée. Cela peut dépendre du type de mouvement de la caméra ou alors de la connaissance qu'on en a *a priori*. On peut également ajouter une contrainte d'unicité afin d'être sûr qu'un point $\mathbf{p}^1$ de l'image 1 n'est apparié qu'avec un seul et unique point de l'image 2, et inversement. On essaiera bien sûr d'éviter au maximum les erreurs d'appariements. Il y a deux cas d'erreurs : dans le premier, on a un couple de points appariés mais ils ne sont pas des points homologues ("faux appariements"). Dans le second cas, on aurait dû trouver un couple de points mais on l'a manqué. Dans tous les cas, les erreurs d'appariements (erronés ou manquants) peuvent faire échouer les calculs géométriques. Cependant, des méthodes robustes aux données aberrantes permettent de limiter les problèmes dûs aux faux appariements (premier cas).

1.2.4 Formulation du problème

Considérons une séquence de m images I^1 à I^m dans laquelle on observe un ensemble de n points 3D $\mathbf{P}_1$ à $\mathbf{P}_n$. Chaque point $\mathbf{P}_j$ peut être vu dans une ou plusieurs images. On fait l'hypothèse dans ce paragraphe que l'étape de mise en correspondance a été réalisée et qu'il n'y a pas de correspondance de points erronée. On note $\mathbf{p}_j^i$ l'observation du j-ème point dans la i-ème image. Si l'on considère que le point j est détecté dans k_j images dont les indices sont notés $i_j^1, i_j^2, \dots i_j^{k_j}$, alors on dispose de k_j observations $\mathbf{p}_j^{i_j^1}, \mathbf{p}_j^{i_j^2}, \dots \mathbf{p}_j^{i_j^{k_j}}$ pour ce point j. Le problème consiste ensuite à calculer, à partir des observations, les coordonnées de tous les points $\mathbf{P}_1$ à $\mathbf{P}_n$ ainsi que les poses des caméras C^1 à C^m correspondantes aux images I^1 à I^m.

Entrées:

m images :	$I^1, I^2, \dots I^m$
n points observés :	$\mathbf{P_1} : \mathbf{p}_1^{i_1^1}, \mathbf{p}_1^{i_1^2}, \dots \mathbf{p}_1^{i_1^{k_1}}$
	$\mathbf{P_2} : \mathbf{p}_2^{i_2^1}, \mathbf{p}_2^{i_2^2}, \dots \mathbf{p}_2^{i_2^{k_2}}$
	$\vdots$
	$\mathbf{P_n} : \mathbf{p}_n^{i_n^1}, \mathbf{p}_n^{i_n^2}, \dots \mathbf{p}_n^{i_n^{k_n}}$

Sorties:

m poses de caméra :	$C^1, C^2, \dots C^m = (\mathtt{R}^1, \mathbf{t}^1), (\mathtt{R}^2, \mathbf{t}^2), \dots (\mathtt{R}^m, \mathbf{t}^m)$
n points 3D :	$\mathbf{P}_1, \mathbf{P}_2, \dots \mathbf{P}_n = (x_1, y_1, z_1), (x_2, y_2, z_2), \dots (x_n, y_n, z_n)$

1.3 Calculs de la géométrie 3D (méthodes directes)

Cette section traite de l'obtention des informations tridimensionnelles relatives aux images. Cela concerne essentiellement l'estimation des paramètres géométriques (points et poses de caméras) qui sont estimés de manière "directe" par la résolution de systèmes linéaires ou polynomiaux.

1.3.1 La géométrie de deux images

On s'intéresse ici à la géométrie liée à l'observation d'une même scène à travers deux points de vue différents. Cette géométrie, introduite par une paire de caméras et indépendante de la structure de la scène, est appelée géométrie épipolaire [19].

La géométrie épipolaire

Considérons un point 3D $\mathbf{P}$ de l'espace se projetant en un point $\mathbf{p}^0$ dans l'image 0 et en un point $\mathbf{p}^1$ dans l'image 1. La géométrie épipolaire décrit la relation qui lie $\mathbf{p}^0$ à $\mathbf{p}^1$ (dits points homologues) sans pour autant connaître $\mathbf{P}$. Comme on peut le voir sur la Figure 1.7, les points projetés $\mathbf{p}^0$ et $\mathbf{p}^1$, les centres de projection $\mathbf{C}^0$ et $\mathbf{C}^1$ des caméras et le point 3D $\mathbf{P}$ sont coplanaires et appartiennent au plan épipolaire noté $\boldsymbol{\pi}$. L'intersection de $\boldsymbol{\pi}$ avec les deux plans image donne les droites épipolaires $\mathbf{l}^0$ et $\mathbf{l}^1$.

Ainsi, le plan défini par $\mathbf{C}^0$, $\mathbf{C}^1$ et $\mathbf{p}^0$ contient aussi $\mathbf{p}^1$. Ce plan intersecte le plan image 1 en une droite, où $\mathbf{p}^1$ doit nécessairement se trouver. La projection de $\mathbf{C}^0$ dans l'image 1, notée $\mathbf{e}^1$ est appelée épipôle. De la même manière, le correspondant $\mathbf{p}^0$ de $\mathbf{p}^1$ dans l'image 0 doit appartenir à la droite épipolaire $\mathbf{l}^0$.

La matrice fondamentale

La matrice fondamentale $\mathtt{F}$ permet de traduire de manière algébrique la géométrie épipolaire. En effet, deux points homologues $\bar{\mathbf{p}}^0 = [x^0 \; y^0 \; 1]^\top$ et $\bar{\mathbf{p}}^1 = [x^1 \; y^1 \; 1]^\top$ sont liés par la relation bilinéaire

$$\bar{\mathbf{p}}^{0^\top} \; \mathtt{F} \; \bar{\mathbf{p}}^1 = 0. \tag{1.11}$$

C'est une condition nécessaire pour que les deux points $\mathbf{p}^0$ et $\mathbf{p}^1$ correspondent aux projections du même point 3D. $\mathtt{F}$ est une matrice 3×3 de rang

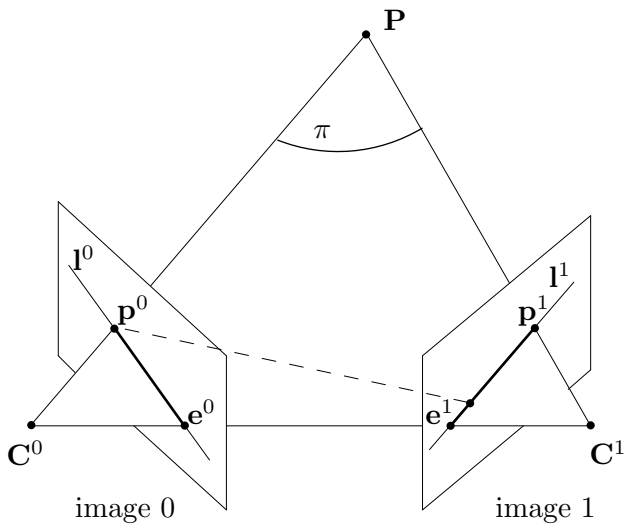


FIG. 1.7 – La géométrie épipolaire.

2 (son déterminant est nul) dépendant à la fois des paramètres intrinsèques et extrinsèques, et définie à un coefficient multiplicatif près. Le nombre de ses paramètres indépendants est donc 7. Cela signifie qu'il faut au moins 7 correspondances de points, donnant chacune une équation $\bar{\mathbf{p}}^{0\top}\ \mathtt{F}\ \bar{\mathbf{p}}^{1} = 0$ pour calculer $\mathtt{F}$. Chaque correspondance engendre donc une équation linéaire qui peut s'écrire sous la forme

$$\mathbf{a}^{\top}\mathbf{f} = 0 \tag{1.12}$$

où $\mathbf{f}$ est un vecteur de taille 9×1 contenant les coefficients de $\mathtt{F}$ tel que

$$\mathbf{f} = [\mathtt{F}_{00}\ \mathtt{F}_{01}\ \mathtt{F}_{02}\ \mathtt{F}_{10}\ \mathtt{F}_{11}\ \mathtt{F}_{12}\ \mathtt{F}_{20}\ \mathtt{F}_{21}\ \mathtt{F}_{22}]^{\top} \tag{1.13}$$

et $\mathbf{a}$ contient des combinaisons des coordonnées de $\mathbf{p}^0$ et $\mathbf{p}^1$

$$\mathbf{a} = [x^0x^1, x^0y^1, x^0, y^0x^1, y^0y^1, y^0, x^1, y^1, 1]^{\top}. \tag{1.14}$$

Trouver $\mathtt{F}$ revient alors à résoudre le système d'équations linéaires obtenus pour N correspondances de points

$$\mathtt{A}_{N\times 9}\mathbf{f} = 0. \tag{1.15}$$

Le cas minimal : 7 correspondances de points Puisque $\mathtt{F}$ a 7 degrés de liberté, il suffit théoriquement de 7 correspondances de points pour résoudre le système linéaire. Cependant, si l'on dispose de seulement 7 points, on remarque que la matrice $\mathtt{A}_{7\times 9}$ est généralement de rang 7. La solution du

système $\mathtt{A}_{7\times 9}\mathbf{f} = 0$ est l'espace à 2 dimensions de la forme $\alpha\mathbf{f}_1 + (1-\alpha)\mathbf{f}_2$ où $\alpha \in \mathbb{R}$ et $\mathbf{f}_1, \mathbf{f}_2$ forment une base du noyau de $\mathtt{A}_{7\times 9}$. Il suffit d'ajouter la contrainte $det(\mathtt{F}) = 0$ ou plutôt $det\,(\alpha\mathbf{f}_1 + (1-\alpha)\mathbf{f}_2) = 0$ (c'est l'algorithme des 7 points [101, 30]). Puisque $\mathbf{f}_1$ et $\mathbf{f}_2$ sont connus, on obtient une équation polynomiale de degré 3 en α. La résolution de cette équation donne 1 ou 3 solutions réelles pour α, ce qui nous permet d'obtenir 1 ou 3 solutions pour $\mathtt{F}$.

L'algorithme des 8 points Pour qu'une solution unique existe (à un facteur près), le rang de la matrice $\mathtt{A}_{N\times 9}$ doit être au moins égal à 8. Pour cela, on ajoute une correspondance de points (ou plus) supplémentaire. Dans ce cas, le rang de $\mathtt{A}_{8\times 9}$ est 8 ou 9 et on peut facilement déterminer une solution unique sans ajouter de contrainte sur le rang de $\mathtt{F}$. C'est l'algorithme des 8 points initialement présenté par Longuet-Higgins en 1981 [47]. Si le rang est exactement 8, la solution est prise dans le noyau de $\mathtt{A}_{N\times 9}$ (de dimension 1). Par contre, si les données ne sont pas exactes (bruit sur les coordonnées des points), alors le rang est 9 et une solution aux moindres carrés minimisant $||\mathtt{A}_{N\times 9}\mathbf{f}||$doit être choisie. Cependant, afin d'éliminer les instabilités numériques, il est important de normaliser les données [27] avant de construire les équations. Les points de l'image sont transformés de façon à ce que le nuage de points soit centré et que la distance moyenne des points au centre soit égale à 1. Cette normalisation améliore grandement le conditionnement du problème et ainsi la stabilité du résultat.

La matrice essentielle

Si l'on se trouve dans le cas calibré (paramètres intrinsèques connus), la matrice essentielle $\mathtt{E}$ introduite par LONGUET-HIGGINS [47] peut être utilisée à la place de la matrice fondamentale. Contrairement à $\mathtt{F}$, elle ne dépend que des paramètres extrinsèques. On considère la représentation des points homologues $\bar{\mathbf{p}}^0 = [x^0\ y^0\ 1]^\top$ et $\bar{\mathbf{p}}^1 = [x^1\ y^1\ 1]^\top$ par leur coordonnées $\mathbf{d}^0$ et $\mathbf{d}^1$ dites *normalisées* sans l'effet du calibrage intrinsèque, qui sont les directions des rayons optiques $\mathbf{d}^0 = (\mathtt{K}^0)^{-1}\,\bar{\mathbf{p}}^0$ et $\mathbf{d}^1 = (\mathtt{K}^1)^{-1}\,\bar{\mathbf{p}}^1$ dans le repère respectivement lié à la caméra 0 et 1. La relation décrivant la géométrie épipolaire devient

$$\mathbf{d}^{0\top}\mathtt{E}\mathbf{d}^1 = 0 \tag{1.16}$$

où $\mathtt{E} = \mathtt{K}^{0\top}\mathtt{F}\mathtt{K}^1$ est la matrice essentielle. Comme pour la matrice fondamentale, il est possible de calculer $\mathtt{E}$ à partir de correspondances de points entre deux images, à condition de connaître les paramètres intrinsèques des deux

prises de vue. Dans cette thèse, on considère qu'une caméra se déplace tout en conservant les mêmes paramètres intrinsèques. Il est donc tout naturel de passer par le calcul de la matrice essentielle. De la même manière que pour $\mathtt{F}$, trouver $\mathtt{E}$ revient à résoudre le système

$$\mathtt{A'}_{N\times 9}\mathbf{e} = 0 \tag{1.17}$$

où $\mathbf{e}$ est un vecteur de taille 9×1 contenant les coefficients de $\mathtt{E}$ tel que

$$\mathbf{e} = [\mathtt{E}_{00}\ \mathtt{E}_{01}\ \mathtt{E}_{02}\ \mathtt{E}_{10}\ \mathtt{E}_{11}\ \mathtt{E}_{12}\ \mathtt{E}_{20}\ \mathtt{E}_{21}\ \mathtt{E}_{22}]^\top. \tag{1.18}$$

Chacune des lignes du système décrivant une correspondance de points s'écrit

$$\mathbf{a'}^\top\mathbf{e} = 0 \tag{1.19}$$

et $\mathbf{a'}$ contient des combinaisons des coordonnées de $\mathbf{d}^0 = [x_d^0\ y_d^0\ z_d^0]^\top$ et $\mathbf{d}^1 = [x_d^1\ y_d^1\ z_d^1]^\top$

$$\mathbf{a'} = [x_d^0x_d^1, x_d^0y_d^1, x_d^0z_d^1, y_d^0x_d^1, y_d^0y_d^1, y_d^0z_d^1, z_d^0x_d^1, z_d^0y_d^1, z_d^0z_d^1]^\top. \tag{1.20}$$

La matrice essentielle $\mathtt{E}$ contient les informations relatives au déplacement de la caméra 1 par rapport à la caméra 0. A partir de $\mathtt{E}$, on peut calculer le mouvement $(\mathtt{R}, \mathbf{t})$ entre les deux prises de vue (rotation + translation de la caméra).

L'algorithme des 5 points Une méthode efficace et minimale pour calculer la matrice essentielle à partir de correspondances de points n'a été publiée que très récemment par D. Nister [69]. La matrice essentielle a 5 degrés de liberté et il suffit de connaître 5 points homologues pour la calculer, d'où le nom d'algorithme des 5 points. Une variante rapide et robuste a été publiée cette année 2007 par Sarkis *et al.* [85].
Dans l'algorithme des 5 points, la solution du système $\mathtt{A'}_{5\times 9}\mathbf{e} = 0$ est l'espace à 4 dimensions formé par la base $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4)$ du noyau (de taille 4) de $\mathtt{A}_{5\times 9}$. La solution $\mathbf{e}$ doit être de la forme

$$\mathbf{e} = x\mathbf{e}_1 + y\mathbf{e}_2 + z\mathbf{e}_3 + w\mathbf{e}_4 \tag{1.21}$$

où $x, y, z, w \in \mathbb{R}$. $\mathtt{E}$ étant définie à un coefficient multiplicatif près, le coefficient w de $\mathbf{e}_4$ est fixé à 1. Trouver la solution $\mathtt{E}$ revient à déterminer le triplet (x, y, z). Pour cela on rajoute une sixième relation traduisant le fait que $\mathtt{E}$ est une matrice essentielle. En effet, toute matrice essentielle est de rang 2 avec

2 valeurs singulières égales et une valeur singulière nulle [19]. $\mathtt{E}$ doit alors vérifier l'équation suivante :

$$\mathtt{E}\mathtt{E}^\top\mathtt{E} - \frac{1}{2}trace(\mathtt{E}\mathtt{E}^\top)\mathtt{E} = 0. \tag{1.22}$$

En combinant les équations 1.21 et 1.22, on obtient un système non linéaire de 9 équations à 3 inconnues. Chacune de ces neuf équations est une équation polynomiale de degré 3 en x, y et z. La première étape consiste à calculer z. Une fois z trouvé, le calcul de x et y est assez simple. Le calcul n'est pas développé ici, pour plus de détails on se référera au papier de D. NISTER [69] et à la thèse de E. ROYER [82].

Extraction des paramètres extrinsèques à partir de la matrice essentielle Lorsqu'on connaît $\mathtt{E}$, il est possible de calculer la pose de la caméra C^1 relativement à la caméra C^0. Il s'agit de calculer le mouvement relatif $(\mathtt{R}, \mathbf{t})$ entre les deux caméras en prenant le repère lié à la caméra 0 comme repère global. Dans ce cas, la pose de la première caméra est $C^0 = (\mathtt{I}_3, \mathbf{0})$ et la pose de la seconde caméra est $C^1 = (\mathtt{R}, \mathbf{t})$. Avec cette notation, la matrice $\mathtt{E}$ s'écrit

$$\mathtt{E} = [\mathbf{t}]_\times \mathtt{R} \tag{1.23}$$

où $[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$ et $\mathbf{t} = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^\top$.

$[\mathbf{t}]_\times$ est la matrice antisymétrique du produit vectoriel par le vecteur 3D $\mathbf{t}$ (autrement dit, $[\mathbf{t}]_\times \mathbf{x} = \mathbf{t} \wedge \mathbf{x}$).

Il faut alors décomposer $\mathtt{E}$ pour en extraire $\mathtt{R}$ et $\mathbf{t}$. Une décomposition en valeurs singulières (SVD) de $\mathtt{E}$ donne $\mathtt{E} = \mathtt{U}diag(1,1,0)\mathtt{V}^\top$ avec $det(U) > 0$ et $det(V)>0$. En effet, rappelons qu'une propriété de la matrice essentielle est qu'elle a ses deux valeurs singulières non nulles égales. Étant donné $\mathtt{E}$, il existe alors 4 possibilités pour $C^1 = (\mathtt{R}, \mathbf{t})$ qui sont [30] :

$$C^1 = (\mathtt{U}\mathtt{W}\mathtt{V}^\top, +\mathbf{u}_3),\ (\mathtt{U}\mathtt{W}\mathtt{V}^\top, -\mathbf{u}_3),\ (\mathtt{U}\mathtt{W}^\top\mathtt{V}^\top, +\mathbf{u}_3),\ (\mathtt{U}\mathtt{W}^\top\mathtt{V}^\top, -\mathbf{u}_3) \tag{1.24}$$

où $\mathtt{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ et $\mathbf{u_3}$ est la troisième colonne de $\mathtt{U}$. Parmi ces quatre solutions, une seule permet de reconstruire tous les points devant les deux caméras simultanément. C'est cette solution qui est choisie.

1.3.2 Reconstruction 3D d'un nuage de points

La détermination des coordonnées 3D d'un point à partir de ses observations dans (au moins) deux images porte le nom d'intersection ou triangulation. En effet, si les positions des caméras sont connues au moment des prises de vue, alors le problème peut se ramener à une simple intersection de deux rayons de projection (formant un triangle) dans l'espace. En l'absence de bruit, c'est un problème trivial. En revanche, lorsque les mesures des points sont bruitées, alors les rayons ne s'intersectent pas et la solution est plus compliquée. Le problème de triangulation optimale a été étudié par Hartley et Sturm [29]. La méthode la plus répandue dans les travaux de reconstruction 3D et celle que nous avons utilisée s'appelle la méthode du "point milieu". Elle consiste à choisir le point situé au milieu de la perpendiculaire commune, minimisant ainsi la somme des distances aux rayons (Figure 1.8(a)). Dans cette section, elle est expliquée de manière très simple pour la triangulation d'un point à partir de deux ou trois vues. Par la suite, c'est cette méthode qui sera utilisée pour la reconstruction des points 3D.

Reconstruction 3D d'un point à partir de deux vues : méthode du point milieu

On considère les poses des caméras $(\mathtt{R}^0, \mathbf{t}^0)$ et $(\mathtt{R}^1, \mathbf{t}^1)$ connues avec leur centre optique en $\mathbf{t}^0$ et $\mathbf{t}^1$.
Le point à reconstruire est observé en $\bar{\mathbf{p}}^0 = [x^0\ y^0\ 1]^\top$ et $\bar{\mathbf{p}}^1 = [x^1\ y^1\ 1]^\top$ dans les 2 images.
Les vecteurs $\mathbf{d}^0$ et $\mathbf{d}^1$ représentent les directions des rayons partant des centres optiques passant par les points détectés dans les images.
On a alors :

$$\mathbf{d}^0 = \frac{\mathtt{R}^0(\mathtt{K}^0)^{-1}\bar{\mathbf{p}}^0}{\|\mathtt{R}^0(\mathtt{K}^0)^{-1}\bar{\mathbf{p}}^0\|} \text{ et } \mathbf{d}^1 = \frac{\mathtt{R}^1(\mathtt{K}^1)^{-1}\bar{\mathbf{p}}^1}{\|\mathtt{R}^1(\mathtt{K}^1)^{-1}\bar{\mathbf{p}}^1\|}$$

et les deux rayons optiques peuvent être paramétrés de la façon suivante :

$$\mathbf{r}^0(\lambda) = \mathbf{t}^0 + \lambda\mathbf{d}^0 \text{ et } \mathbf{r}^1(\mu) = \mathbf{t}^1 + \mu\mathbf{d}^1. \tag{1.25}$$

Les deux rayons ne sont probablement pas sécants ni parallèles, et il existe une droite $(\mathbf{H}^0\mathbf{H}^1)$ qui leur est orthogonale, $\mathbf{H}^0 \in \mathbf{r}^0$ et $\mathbf{H}^1 \in \mathbf{r}^1$. On cherche le point $\mathbf{P}$ situé au milieu du segment $[\mathbf{H}^0\mathbf{H}^1]$ et on commence par déterminer les valeurs de λ et μ telles que :

$$\mathbf{H}^0 = \mathbf{t}^0 + \lambda\mathbf{d}^0 \text{ et } \mathbf{H}^1 = \mathbf{t}^1 + \mu\mathbf{d}^1 \tag{1.26}$$

Le vecteur $\mathbf{w} = \mathbf{d}^0 \wedge \mathbf{d}^1$ est le support de la droite $(\mathbf{H}^0\mathbf{H}^1)$
On a donc :

$$\mathbf{w} \;//\; (\mathbf{H}^0\mathbf{H}^1) \text{ soit } \mathbf{d}^0 \wedge \mathbf{d}^1 \;//\mathbf{t}^1 - \mathbf{t}^0 + \mu\mathbf{d}^1 - \lambda\mathbf{d}^0$$

ce qui est équivalent à :

$$\begin{cases} \mathbf{d}^0 \cdot \; (\mathbf{t}^1 - \mathbf{t}^0 + \mu\mathbf{d}^1 - \lambda\mathbf{d}^0) \;=\; 0 \\ \mathbf{d}^1 \cdot \; (\mathbf{t}^1 - \mathbf{t}^0 + \mu\mathbf{d}^1 - \lambda\mathbf{d}^0) \;=\; 0 \end{cases} \tag{1.27}$$

Cela permet de déterminer λ et μ :

$$\begin{aligned} \lambda \;&=\; \frac{\| \mathbf{d}^1 \|^2 \times (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^0 \;-\; (\mathbf{d}^0 \cdot \mathbf{d}^1) \times (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^1}{\| \mathbf{d}^0 \|^2 \times \| \mathbf{d}^1 \|^2 - (\mathbf{d}^0 \cdot \mathbf{d}^1)^2} \\ \mu \;&=\; \frac{(\mathbf{d}^0 \cdot \mathbf{d}^1) \times (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^0 \;-\; (\| \mathbf{d}^0 \|^2 \times (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^1}{\| \mathbf{d}^0 \|^2 \times \| \mathbf{d}^1 \|^2 - (\mathbf{d}^0 \cdot \mathbf{d}^1)^2} \end{aligned} \tag{1.28}$$

Puisque les vecteurs $\mathbf{d}^0$ et $\mathbf{d}^1$ sont normés :

$$\begin{aligned} \lambda \;&=\; \frac{(\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^0 \;-\; (\mathbf{d}^0 \cdot \mathbf{d}^1) \times (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^1}{1 - (\mathbf{d}^0 \cdot \mathbf{d}^1)^2} \\ \mu \;&=\; \frac{(\mathbf{d}^0 \cdot \mathbf{d}^1) \times (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^0 \;-\; (\mathbf{t}^1 - \mathbf{t}^0) \cdot \mathbf{d}^1}{1 - (\mathbf{d}^0 \cdot \mathbf{d}^1)^2} \end{aligned} \tag{1.29}$$

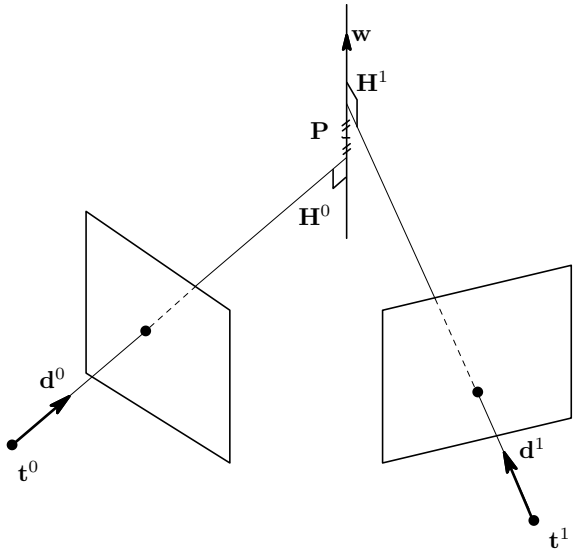
Le point $\mathbf{P}$ est reconstruit au milieu de $[\mathbf{H}^0\mathbf{H}^1]$ tel que

$$\mathbf{P} \;=\; \frac{1}{2}\left[(\mathbf{t}^0 + \lambda\mathbf{d}^0) + (\mathbf{t}^1 + \mu\mathbf{d}^1)\right]. \tag{1.30}$$

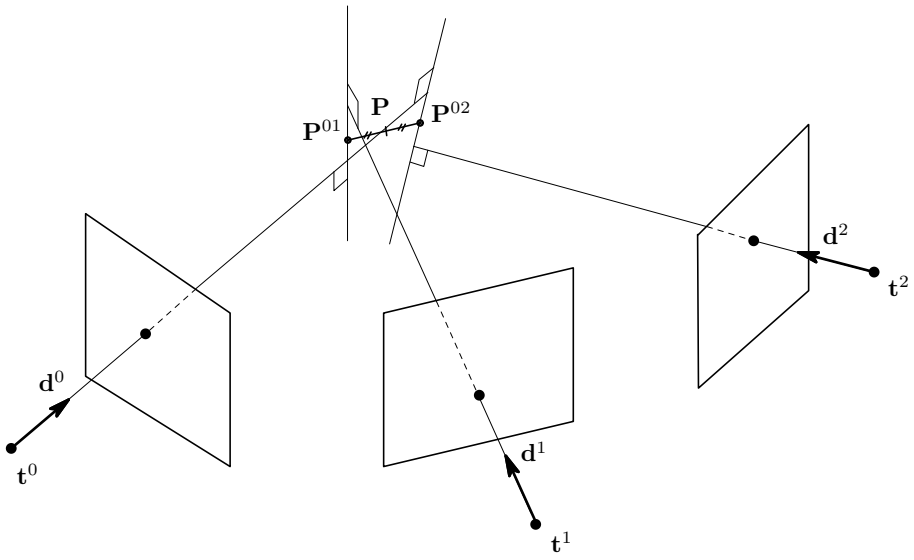
Reconstruction 3D d'un point à partir de trois observations

Dans le travail réalisé, les points ne sont reconstruits qu'à condition d'être observés dans au moins trois prises de vue. On considère les poses des trois caméras connues. Dans ce cas, les coordonnées tridimensionnelles des points sont estimées de la façon suivante, comme illustré sur la Figure 1.8(b) :

- Une première estimation $\mathbf{P}^{01}$ est réalisée à l'aide des 2 premières caméras 0 et 1 avec la méthode précédemment présentée.
- Une seconde estimation $\mathbf{P}^{02}$ est réalisée avec la même méthode en utilisant la première et la dernière caméra afin de maximiser l'angle de triangulation.
- On choisit finalement le milieu $\mathbf{P}$ du segment $[\mathbf{P}^{01}\mathbf{P}^{02}]$.



(a) 2 vues : le point milieu de la perpendiculaire commune aux rayons de projection.



(b) 3 vues : le point milieu du segment formé par les points $\mathbf{P}^{01}$ et $\mathbf{P}^{02}$.

FIG. 1.8 – Détermination d'un point $\mathbf{P}$ dans le cas d'une reconstruction avec (a) 2 vues, (b) 3 vues.

1.3.3 Calcul de pose

L'estimation de la pose d'une caméra consiste à déterminer la position et l'orientation de la caméra (calibrée) à partir de points 3D déjà calculés et

leurs coordonnées 2D dans l'image. Il s'agit donc de déterminer la pose $(\mathtt{R}, \mathbf{t})$ à partir de correspondances 3D/2D : $\mathbf{P}_j \leftrightarrow \mathbf{p}_j$. Ce problème est aussi appelé résection par les photogrammètres.

Calcul de pose à partir de 3 points

Avec 3 points, le problème du calcul de pose possède quatre solutions dans le cas général mais le plus souvent deux solutions. De nombreuses méthodes sont connues pour calculer ces solutions. On attribue souvent à LAGRANGE à la fin du XVIII^{e} siècle les premiers travaux sur le calcul du point de vue perspectif à partir de 3 points mais c'est GRÜNERT qui a explicité les solutions algébriques du problème en 1841 [23]. Depuis, de nombreuses variantes de l'algorithme des 3 points ont été développées. HARALICK *et al.* [24] ont étudié différentes variantes, anciennes et plus récentes, de la méthode utilisant 3 points, et comparé les différences de stabilité numérique.

Les algorithmes étudiés fonctionnent de la manière suivante. Étant donné n correspondances entre les points 3D $\mathbf{P}_j$ $(j = 1 \dots n)$ et leur projection $\mathbf{p}_j$ dans l'image, chaque paire de correspondance $\mathbf{P}_j \leftrightarrow \mathbf{p}_j$ et $\mathbf{P}_k \leftrightarrow \mathbf{p}_k$ donne une contrainte sur les distances inconnues entre les points 3D et le centre de la caméra situé en $\mathbf{t}$: $x_j = ||\mathbf{P}_j - \mathbf{t}||$ et $x_k = ||\mathbf{P}_k - \mathbf{t}||$ (voir Figure 1.9). Chaque contrainte s'écrit :

$$d_{jk}^2 = x_j^2 + x_k^2 - 2x_j x_k \ \cos\theta_{jk} \tag{1.31}$$

où $d_{jk} = ||\mathbf{P}_j - \mathbf{P}_k||$ est la distance connue entre les points 3D $\mathbf{P}_j$ et $\mathbf{P}_k$, et θ_{jk} est l'angle entre les deux rayons optiques partant du centre optique en $\mathbf{t}$ et passant par les points $\mathbf{p}_j$ et $\mathbf{p}_k$. Leur direction dans le repère lié à la caméra est donnée par les vecteurs normés $\mathbf{d}_j$ et $\mathbf{d}_k$ tels que $\cos\theta_{jk} = \mathbf{d}_j \cdot \mathbf{d}_j$ avec

$$\mathbf{d}_j = \frac{\mathtt{K}^{-1}\bar{\mathbf{p}}_j}{||\mathtt{K}^{-1}\bar{\mathbf{p}}_j||} \text{ et } \mathbf{d}_k = \frac{\mathtt{K}^{-1}\bar{\mathbf{p}}_k}{||\mathtt{K}^{-1}\bar{\mathbf{p}}_k||}. \tag{1.32}$$

La contrainte peut s'écrire de la façon suivante :

$$f_{jk}(x_j, x_k) = x_j^2 + x_k^2 - 2x_j x_k \cos\theta_{jk} - d_{jk}^2 = 0. \tag{1.33}$$

Pour $n = 3$ points, on obtient le système d'équations quadratiques suivant :

$$\begin{cases} f_{01}(x_0, x_1) = 0 \\ f_{02}(x_0, x_2) = 0 \\ f_{12}(x_1, x_2) = 0 \end{cases} \tag{1.34}$$

où les trois inconnues sont x_0, x_1 et x_2. Ensuite, les méthodes de résolution sont différentes mais très proches les unes des autres. Celle de GRÜNERT [23] utilise les changements de variables $x_1 = ux_0$ et $x_2 = vx_0$ pour réduire le nombre d'inconnues aux deux variables u et v. On peut ensuite éliminer u par substitution afin d'obtenir un polynôme de degré 4 en v :

$$g(v) = A_4v^4 + A_3v^3 + A_2v^2 + A_1v + A_0 = 0. \tag{1.35}$$

Une telle équation a au plus 4 solutions en v et peut être résolue de manière explicite. Chaque solution pour v donne une solution pour u. Une fois u et v connus, il est alors facile de déterminer x_0, et le calcul des valeurs de x_1 et x_2 est immédiat à partir des changements de variables.

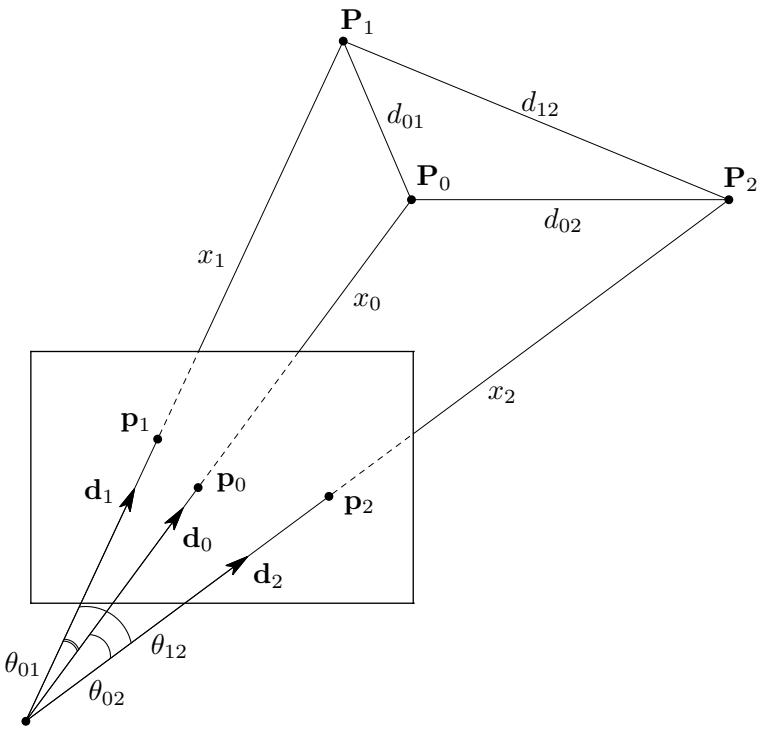


FIG. 1.9 – Détermination de la pose : contrainte géométrique pour trois correspondances 3D/2D : $\mathbf{P}_0 \leftrightarrow \mathbf{p}_0$, $\mathbf{P}_1 \leftrightarrow \mathbf{p}_1$ et $\mathbf{P}_2 \leftrightarrow \mathbf{p}_2$.

Calcul de pose à partir de n ($n > 3$) points

La méthode de 3 points possède des solutions multiples. Si l'on veut obtenir une solution unique, il faut ajouter un quatrième point. Une approche consiste à prendre des triplets de points parmi les quatre, de résoudre le

polynôme de degré 4 pour chaque triplet et de choisir finalement la solution commune. Cette méthode est très utilisée, mais la solution unique est difficile à déterminer si les données sont bruitées et elle ne prend pas en compte les redondances si plus de quatre points sont disponibles. Le calcul de la pose d'une caméra à partir de données redondantes a également été très étudié. Une grande difficulté est de prendre en compte le bruit et les fausses données. Les algorithmes linéaires [76] sont rapides mais ils sont sensibles au bruit et peuvent présenter des instabilités numériques.
On peut alors rechercher une solution à partir d'un nombre réduit de points et calculer ensuite une solution optimale (voir section suivante). Dans ce sens, FISCHLER et BOLLES [21] ont introduit une méthode nommée "RANSAC" (expliquée en 1.4.4) qui fonctionne par tirages aléatoires afin d'éliminer les valeurs aberrantes dans les correspondances de points. La plupart des méthodes qui utilisent ce principe sont itératives et optimisent une estimation initiale approximative fournie par RANSAC. Ces méthodes, stables numériquement, peuvent poser des problèmes car la convergence dépend fortement de l'estimation initiale. Par exemple, LOWE [50] et YUAN [104] ont développé des techniques basées sur la méthode de NEWTON-RAPHSON. DEMENTHON et DAVIS [14] présentent quant à eux une méthode itérative pour quatre points et plus, initialisée à partir d'une approximation orthographique.

1.4 Solutions optimales (méthodes itératives)

La reconstruction 3D commence par une estimation des différents paramètres : poses de la caméra à différents instants et coordonnées 3D des points de l'espace. Cependant, le système à résoudre est souvent trop contraint, c'est-à-dire que l'on dispose en général de plus d'équations qu'il n'y a d'inconnues. Dans notre cas, cela signifie que l'on dispose, soit de plus de correspondances 3D/2D que nécessaire pour calculer la pose d'une caméra, soit de plus d'observations 2D que nécessaire pour trianguler un point. Si les données sont exactes et sans bruit, alors toutes les combinaisons minimales d'équations permettant de résoudre le problème sous-jacent, donnent la même solution. Par contre, dans le cas réel, le bruit dans les images et les imprécisions de calcul font qu'une solution calculée à partir d'un nombre minimal d'équations n'est pas solution pour une autre combinaison d'équations. En d'autres termes, si l'on dispose de 100 correspondances de points 3D/2D pour calculer la pose d'une caméra et que seulement 3 points sont nécessaires pour ce calcul, alors chaque combinaison de 3 points pris parmi les 100 donnent

une solution. Idéalement, chaque tirage devrait donner la même pose mais ce n'est jamais le cas dans la réalité, et les données aberrantes peuvent produire des solutions complètement fausses. Une méthode itérative est alors utile pour deux raisons : obtenir une estimation qui soit (1) statistiquement optimale pour l'ensemble des mesures et (2) robuste aux mesures aberrantes.

1.4.1 Moindres carrés non-linéaires

Le problème que nous cherchons à résoudre est celui de l'estimation d'un certain nombre de paramètres à partir d'une ou plusieurs images. Supposons que l'ensemble des paramètres à estimer soit contenu dans un vecteur $\mathbf{X}$. La fonction de modélisation non-linéaire f nous donne la relation entre les paramètres inconnus $\mathbf{X}$ et un vecteur de mesure noté $\mathbf{Y}$. Dans le cas idéal, le modèle calculé est en parfaite adéquation avec les mesures et l'on a :

$$f(\mathbf{X}) = \mathbf{Y}. \tag{1.36}$$

En réalité, l'égalité n'est pas parfaite à cause des imprécisions dans les calculs et le bruit de mesure. Pour chaque mesure i, il existe une erreur ϵ_i et on définit le vecteur des résidus

$$\boldsymbol{\epsilon} = f(\mathbf{X}) - \mathbf{Y} \tag{1.37}$$

comme le vecteur contenant l'ensemble des erreurs. Le problème est de retrouver les valeurs de $\mathbf{X}$ à partir des mesures contenues dans $\mathbf{Y}$. La méthode la plus utilisée est la méthode des moindres carrés LS (Least Squares). On cherche à minimiser la fonction suivante (dite fonction de coût) :

$$C(\mathbf{X}) = ||\boldsymbol{\epsilon}||^2 = \boldsymbol{\epsilon}^\top \boldsymbol{\epsilon} = \sum_i ||\epsilon_i||^2. \tag{1.38}$$

La solution de ce problème d'optimisation peut être obtenue de manière itérative en partant d'une estimation initiale des paramètres donnée dans la section 1.5. Si le modèle d'erreur est gaussien (plus précisément, si $\boldsymbol{\epsilon}$ suit une loi normale centrée et réduite), alors la solution aux moindres carrés correspond à l'estimation du maximum de vraisemblance. En ce sens statistique, l'estimation obtenue est "optimale".

1.4.2 L'erreur de reprojection

Dans le processus de reconstruction 3D, les paramètres à estimer sont les coordonnées 3D (x, y, z) des points observés et la pose des caméras $(\mathtt{R}, \mathbf{t})$.

Les mesures sont les valeurs en pixels des observations des points dans les images. L'adéquation entre les mesures et le modèle 3D calculé peut être mesurée grâce à l'erreur de reprojection 2D dans l'image. Soit un point 3D $\mathbf{P}$ et une caméra C dont la matrice de projection est $\mathtt{C}$. Le point $\mathbf{P}$ est détecté au point $\mathbf{p}$ dans l'image mais il ne se projette pas exactement en $\mathbf{p}$. On peut mesurer l'erreur de reprojection :

$$\epsilon = \|\mathbf{p} - \pi\left(\mathtt{C}\bar{\mathbf{P}}\right)\|. \tag{1.39}$$

ϵ est la distance euclidienne dans le plan image entre le point détecté et le point projeté.
L'erreur de reprojection correspondant au j-ème point par la i-ème caméra est noté ϵ_j^i.
En général, si l'on dispose de n observations, alors la qualité de l'estimation du modèle 3D peut être mesurée par le coefficient RMS (Root Mean Square) exprimé en pixels tel que

$$RMS = \sqrt{\frac{1}{n}\sum_i \sum_j ||\epsilon_j^i||^2}. \tag{1.40}$$

1.4.3 Intersection - Résection - Ajustement de faisceaux

Intersection

Si l'on se place dans le contexte de l'estimation des coordonnées d'un point 3D $\mathbf{P}_j$ à partir des ses observations dans les images $I^1, \ldots, I^m$(dont les paramètres du système de prise de vues sont parfaitement connus) alors le vecteur $\mathbf{X}$ des paramètres à estimer est directement égal aux coordonnées de $\mathbf{P}_j$ (voir Figure 1.10). L'estimation optimale au sens des moindres carrés est obtenue en prenant le minimum de la fonction

$$f_j(\mathbf{P}_j) = \sum_{i=1}^{m} ||\epsilon_j^i||^2. \tag{1.41}$$

Les ϵ_j^i représentent les erreurs de reprojection du point $\mathbf{P}_j$ dans les images $I^1, \ldots, I^m$.

Résection

D'une manière équivalente, lors du calcul de la pose de la caméra C^i correspondante à l'image I^i à partir de n points 3D et de leurs observations

$\mathbf{p}_1, \ldots, \mathbf{p}_n$, l'ensemble des paramètres à estimer est réduit aux paramètres $(\mathtt{R}^i, \mathbf{t}^i)$(voir Figure 1.10). La pose optimale est obtenue en minimisant :

$$f^i(C^i) = \sum_{j=1}^{n} ||\epsilon_j^i||^2. \tag{1.42}$$

Les ϵ_j^i représentent les erreurs de reprojection des points $\mathbf{P}_1, \ldots, \mathbf{P}_n$ dans l'image I^i.

Ajustement de faisceaux

On appelle ajustement de faisceaux ("bundle adjustment" en anglais) l'étape de calcul d'une solution optimale où le vecteur $\mathbf{X}$ des paramètres recherchés contient à la fois les poses d'un ensemble $\mathcal{C} = \left\{C^1, \ldots, C^{N_c}\right\}$ de N_c caméras et les coordonnées d'un ensemble $\mathcal{P} = \left\{\mathbf{P}_1, \ldots, \mathbf{P}_{N_p}\right\}$ de N_p points 3D. La fonction à minimiser est

$$f(\mathcal{P}, \mathcal{C}) = \sum_i \sum_j ||\epsilon_j^i||^2 \tag{1.43}$$

où chaque ϵ_j^i mesure l'erreur de reprojection du point $\mathbf{P}_j$ dans l'image I^i.
Remarque : Les points 3D ne sont pas toujours visibles dans toutes les images et les ϵ_j^i ne sont pas nécessairement définis pour tous les $(i, j) \in \{1, \ldots, N_c\} \times \{1, \ldots, N_p\}$.

Dans les trois cas (intersection, résection, et ajustement de faisceaux), la solution peut être calculée grâce à une minimisation non-linéaire (par exemple par les algorithmes de Newton-Raphson ou Levenberg-Marquardt, etc) de la fonction de coût. Cette étape de minimisation n'est pas expliquée ici mais dans la prochaine section 1.5 sur l'ajustement de faisceaux.

1.4.4 Méthodes robustes

L'estimateur des moindres carrés est dit "non-robuste". En effet, un tel estimateur est extrêmement sensible aux mesures aberrantes qui ont un rôle prépondérant dans la minimisation du critère de reprojection. Des alternatives sont proposées pour éviter ces problèmes. Tout d'abord, il y a les estimateurs type LMS (Least Median of Squares) [80, 81] ou LTS (Least Trimmed Squares) [81] qui minimisent respectivement la valeur médiane des résidus et la somme des carrés d'un sous-ensemble de résidus (les plus petits). Les

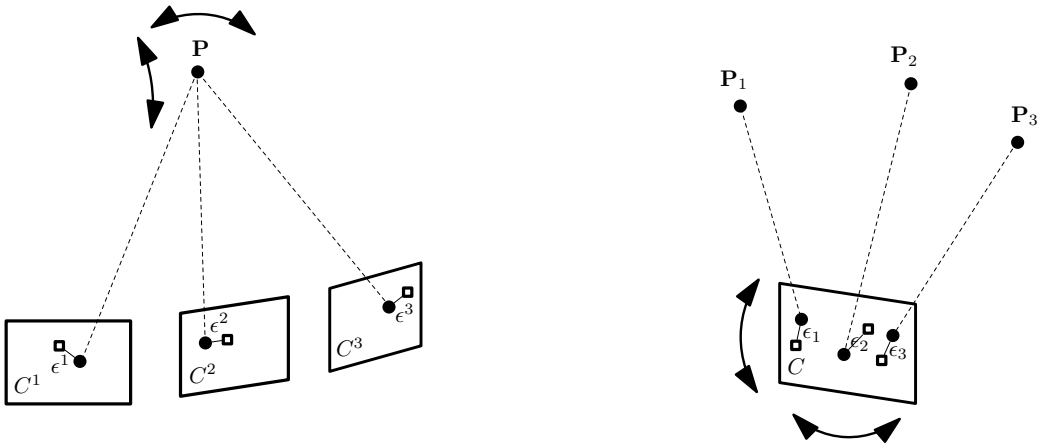


FIG. 1.10 – Intersection/Résection.

deux fonctions de coût ne sont pas différentiables et sont donc plus difficiles à mettre en oeuvre avec des méthodes de minimisation non-linéaires. Les prochains paragraphes expliquent deux solutions pour réduire l'influence des points aberrants : d'abord en choisissant une erreur de mesure robuste, ensuite en éliminant les faux points par une méthode robuste (RANSAC).

Erreur de reprojection robuste

Le principe est de modifier la fonction de coût des moindres carrés en traitant de manière différente les résidus les plus grands. L'erreur ϵ est remplacée par une erreur robuste ϵ_r telle que $\epsilon_r = \rho(\epsilon)$. Dans la littérature, on parle de *M-estimateur* et différentes fonctions ρ ont été proposées. Les plus utilisées sont celles proposées par TUKEY [103] et HUBER [32].

La fonction proposée par HUBER est la suivante :

$$\rho(\epsilon) = \begin{cases} \dfrac{1}{2}\epsilon^2 & \text{si } ||\epsilon|| \leq c \\ c\left(|\epsilon| - \dfrac{c}{2}\right) & \text{si } ||\epsilon|| > c \end{cases} \tag{1.44}$$

La fonction de TUKEY est :

$$\rho(\epsilon) = \begin{cases} \dfrac{c^2}{6}\left[1 - \left(1 - \left(\dfrac{\epsilon}{c}\right)^2\right)^3\right] & \text{si } ||\epsilon|| \leq c \\ \dfrac{c^2}{6} & \text{si } ||\epsilon|| > c \end{cases} \tag{1.45}$$

où c est une constante déterminée à partir de l'estimation de l'écart type du bruit sur les mesures. Dans les deux cas, si les résidus sont faibles, alors ils

sont considérés comme dans les moindres carrés. Par contre, si ils sont importants, leur influence est très limitée dans la fonction de HUBER et totalement annulée dans la fonction de TUKEY (dérivée nulle). La principale conséquence est un ralentissement de la vitesse de convergence des algorithmes d'optimisation. En effet, il est très difficile (voire impossible) de distinguer dans un premier temps entre les mesures aberrantes et les mesures correctes. Donc, des résidus corrects importants sont aussi initialement filtrés, ce qui conduit à une réduction de la vitesse de convergence.

Plus récemment, ENGELS *et al.* [17] proposent une méthode comparable à celle présentée au chapitre 3 dans laquelle l'erreur robuste $\rho(\epsilon) = \ln\left(1 + \frac{\epsilon^2}{\sigma^2}\right)$ est utilisée. Ils font l'hypothèse que les erreurs de reprojection suivent une distribution de CAUCHY d'écart type σ.

Le processus RANSAC (Random Sample Concensus)

L'algorithme RANSAC [21] est un algorithme robuste permettant de faire correspondre des mesures à un modèle en présence de données aberrantes. A l'origine, il a été développé afin de réduire le temps de calcul des méthodes de votes comme la transformée de HOUGH [31]. L'algorithme est très simple : il s'agit de déterminer les paramètres $\mathbf{X}$ à partir d'un ensemble $\mathbf{Y}$ contenant M observations. Cependant, il suffit de m ($m \leq M$) données pour déterminer $\mathbf{X}$. On procède alors par tirages aléatoires en prenant des échantillons de m valeurs dans $\mathbf{Y}$. Chaque tirage t donne un sous-ensemble de données $\mathbf{y}_t$ permettant chacun de déterminer une solution $\mathbf{X}_t$ pour $\mathbf{X}$. Pour chaque solution obtenue $\mathbf{X}_t$, on compte le nombre k_t de mesures prises dans $\mathbf{Y}$ qui valident le modèle calculé (on introduit la notion d'*inliers*). Pour cela, il est nécessaire de définir une fonction d'erreur et un seuil au-dessus duquel les données ne sont pas considérées comme acceptables. On choisit la première solution pour laquelle k_t dépasse un nombre fixé k_s ou alors la solution qui maximise k_t à partir d'un nombre de tirages donné. Le nombre de tirage préconisé dépend de la proportion estimée de mesures aberrantes (on parle d'*outliers*).

L'algorithme RANSAC est très utilisé en vision par ordinateur, par exemple pour le calcul de pose ou le calcul de la géométrie épipolaire où de nombreuses correspondances de points sont disponibles (jusqu'à plusieurs centaines) alors que le calcul en nécessite très peu : 3 pour le calcul de pose ou 5 pour la matrice essentielle.

1.5 Ajustement de faisceaux par minimisation non-linéaire

1.5.1 Introduction

Une fois que l'on dispose d'une estimation initiale de la géométrie pour tout ou une partie de la séquence vidéo, il est recommandé de raffiner les paramètres de ce modèle (position des points + pose des caméras) à travers une étape d'optimisation. Une solution optimale peut être calculée par ajustement de faisceaux. C'est une technique mise au point par les photogrammètres et qui s'est largement répandue dans le domaine de la vision par ordinateur. De nombreux travaux sur le SfM l'utilisent ([73, 26, 5, 22] et d'autres) pour l'ultime étape des algorithmes de reconstruction. L'objectif est de re-estimer simultanément les coordonnées des points 3D et les poses des caméras afin de minimiser la somme des carrés des écarts entre les points détectés dans les images et les reprojections obtenues à partir du modèle calculé. La fonction de coût minimisée est :

$$\sum_i \sum_j ||\epsilon_j^i||^2 = \sum_i \sum_j ||\mathbf{p}_j^i - \pi\left(\mathtt{C}^i \bar{\mathbf{P}}_j\right)||^2 \tag{1.46}$$

où $||\epsilon_j^i||^2$ est le carré de la distance euclidienne entre $\mathtt{C}^i\bar{\mathbf{P}}_j$, la projection estimée du j-ème point 3D $\mathbf{P}_j$ à travers la i-ème caméra calculée C^i et le point détecté $\mathbf{p}_j^i$ correspondant. $\mathtt{C}^i$ est la matrice de projection 3×4 de la caméra i composée des paramètres extrinsèques de C^i et des paramètres intrinsèques connus.

Un état de l'art des algorithmes d'ajustement de faisceaux a été publié en 2000 par TRIGGS *et al.* [102] et un "package" logiciel sous licence GNU est mis à disposition par l'équipe de LOURAKIS et ARGYROS [48].

1.5.2 Formulation

Étant donné un ensemble de points et de caméras dont les paramètres 3D ont été initialement estimés, l'ajustement de faisceaux consiste à améliorer ces paramètres afin d'augmenter l'adéquation entre le modèle 3D et les données mesurées dans les images. Il s'agit donc dans notre cas de trouver les positions des points 3D et les poses des caméras (position+orientation) qui minimisent l'erreur de reprojection dans les images.
On note $\mathbf{X}$ le vecteur contenant l'ensemble des paramètres des points et caméras à re-estimer. Il est de taille $N = 3 \times N_p + 6 \times N_c$ où N_p et N_c sont

respectivement le nombre de points et de caméras impliqués dans l'ajustement de faisceaux. Chaque point a 3 paramètres (coordonnées 3D) et chaque caméra a 3 paramètres qui correspondent à la position 3D (t_x, t_y, t_z) du centre optique et 3 paramètres (α, β, γ) qui correspondent à son orientation. On note $\mathbf{Y}$ le vecteur de mesure de taille M. Dans le cas idéal, on a $\mathbf{Y} = f(\mathbf{X})$ où $f : \mathbb{R}^N \rightarrow \mathbb{R}^M$ est la fonction de projection. Cependant, les erreurs d'estimation et le bruit dans les images font que la relation n'est jamais exactement vérifiée. On cherche donc le vecteur de paramètres $\hat{\mathbf{X}}$ pour lequel l'erreur $||\boldsymbol{\epsilon}|| = ||\mathbf{Y} - f(\hat{\mathbf{X}})||$ est minimale, c'est-à-dire :

$$\hat{\mathbf{X}} = argmin_{\mathbf{X} \in \mathbb{R}^N} ||\mathbf{Y} - f(\mathbf{X})||. \tag{1.47}$$

C'est un problème de moindres carrés non-linéaires et la recherche de $\hat{\mathbf{X}}$ est itérative en faisant l'hypothèse que f est localement linéaire. A chaque itération i, on calcule le pas (ou incrément) $\boldsymbol{\Delta}_i \in \mathbb{R}^N$ à appliquer au vecteur de paramètres $\mathbf{X}_i$ tel que $\mathbf{X}_{i+1} = \mathbf{X}_i + \boldsymbol{\Delta}_i$ en vérifiant que $||\mathbf{Y} - f(\mathbf{X}_{i+1})|| < ||\mathbf{Y} - f(\mathbf{X}_i)||$ ce qui assure la décroissance de l'erreur. En partant de l'estimation initiale $\mathbf{X}_0$, on obtient une série $\mathbf{X}_0,\ \mathbf{X}_1,\ \mathbf{X}_2, \ldots \mathbf{X}_n$ qui converge vers un minimum local $\hat{\mathbf{X}}$ de f.
Pour chaque $\boldsymbol{\Delta}_i$, on a $\boldsymbol{\Delta}_i = \alpha_i \mathbf{D}_i$ où $\mathbf{D}_i$ est la direction de descente et α_i la longueur du pas pour l'itération i. Il existe plusieurs stratégies dans le choix de la direction de descente $\mathbf{D}_i$ au point $\mathbf{X}_i$.

1.5.3 L'itération de Gauss-Newton

L'approximation de Taylor au premier ordre de f en $\mathbf{X}$ donne $f(\mathbf{X} + \boldsymbol{\Delta}) \approx f(\mathbf{X}) + \mathtt{J}\boldsymbol{\Delta}$ où $\mathtt{J}$ est la matrice Jacobienne de f en $\mathbf{X}$ et $||\boldsymbol{\Delta}||$ est petit. On cherche un pas $\boldsymbol{\Delta}$ qui minimise la quantité

$$||\mathbf{Y} - f(\mathbf{X} + \boldsymbol{\Delta})|| \approx ||\mathbf{Y} - f(\mathbf{X}) - \mathtt{J}\boldsymbol{\Delta}|| = ||\boldsymbol{\epsilon} - \mathtt{J}\boldsymbol{\Delta}||. \tag{1.48}$$

Le problème à résoudre revient à un problème de moindres carrés linéaires et consiste à trouver $\boldsymbol{\Delta}$ pour lequel $||\boldsymbol{\epsilon} - \mathtt{J}\boldsymbol{\Delta}||$ est minimal. Le minimum est atteint lorsque $\mathtt{J}\boldsymbol{\Delta} - \boldsymbol{\epsilon}$ est orthogonal à l'espace colonne de $\mathtt{J}$ ce qui revient à résoudre $\mathtt{J}^\top(\mathtt{J}\boldsymbol{\Delta} - \boldsymbol{\epsilon}) = \mathtt{0}$. On obtient alors le pas de Gauss-Newton noté $\boldsymbol{\Delta}_{GN}$ par résolution des "équations normales"

$$\mathtt{J}^\top \mathtt{J} \boldsymbol{\Delta}_{GN} = \mathtt{J}^\top \boldsymbol{\epsilon}. \tag{1.49}$$

On calcule alors les itérations successives de Gauss-Newton jusqu'à convergence. Malheureusement, il arrive que le minimum obtenu soit un minimum local différent de la solution recherchée $\hat{\mathbf{X}}$ ou même que l'algorithme

diverge. Le comportement et la convergence dépendent fortement de l'estimation initiale $\mathbf{X}_0$.

1.5.4 L'itération de descente de Gradient

Avec la méthode de descente de gradient (ou de la plus profonde descente), on choisit la direction de descente $\mathbf{D}$ dans le sens de l'antigradient $\mathbf{g} = \mathtt{J}^\top \boldsymbol{\epsilon}$. En effet, le gradient de $\frac{1}{2}||\boldsymbol{\epsilon}||^2 = \frac{1}{2}\boldsymbol{\epsilon}^\top\boldsymbol{\epsilon}$ est $-\mathtt{J}^\top\boldsymbol{\epsilon}$. La longueur du pas α est à déterminer et l'on peut prendre le point de CAUCHY qui minimise le modèle quadratique de $||\boldsymbol{\epsilon}||^2$. Dans ce cas, le pas $\boldsymbol{\Delta}_{DG}$ de descente de gradient est

$$\boldsymbol{\Delta}_{DG} = \frac{||\mathbf{g}||^2}{||\mathtt{J}\mathbf{g}||^2}\mathbf{g}. \tag{1.50}$$

En pratique, on observe souvent que $\mathbf{D} = \mathbf{g}$ est une bonne direction de descente loin d'une solution mais qu'elle est à éviter dès que l'on entre dans le voisinage d'une solution $\hat{\mathbf{X}}$, là où les termes du second ordre d'un développement de TAYLOR de f autour de $\hat{\mathbf{X}}$ jouent un grand rôle. En fait, le défaut de cette méthode de descente est d'ignorer la courbure de f en $\mathbf{X}$, qui est décrite par son Hessien approché $\mathtt{J}^\top\mathtt{J}$.

1.5.5 L'algorithme de Levenberg-Marquardt

L'algorithme publié en premier par K. LEVENBERG [41] puis redécouvert par D. MARQUARDT [54] est un algorithme itératif qui permet d'obtenir une solution numérique au problème de minimisation d'une fonction dépendante de plusieurs variables. Cette méthode est devenue un standard parmi les méthodes itératives d'optimisation non-linéaire. Chaque itération combine la méthode de GAUSS-NEWTON et la méthode de descente de gradient. Quand la solution est loin du minimum local, on privilégie la descente de gradient qui offre une convergence lente mais assurée. Au contraire, quand on s'approche du minimum local, alors c'est la méthode de GAUSS-NEWTON qui est mise en avant et permet une convergence plus efficace.
Dans l'algorithme de LEVENBERG-MARQUARDT [30, 75] on résout une équation très peu différente de l'équation (1.49). En fait, les termes diagonaux de la matrice $\mathtt{N} = \mathtt{J}^\top\mathtt{J}$ sont multipliés par $(1+\lambda)$, où $\lambda \in \mathbb{R}$ et $\lambda > 0$. Cela revient à résoudre :

$$\mathtt{N}'\boldsymbol{\Delta}_{LM} = \mathtt{J}^\top\boldsymbol{\epsilon} \tag{1.51}$$

avec $\mathtt{N}'(i,i) = (1+\lambda)\mathtt{N}(i,i)\ \forall\ i$ et $\mathtt{N}'(i,j) = \mathtt{N}(i,j)$ pour $i \neq j$.
Le fait de modifier ainsi la diagonale de la matrice est appelé "amortisse-

ment" ou "damping" en anglais, et le paramètre λ est dit "coefficient d'amortissement". Augmenter (resp. réduire) λ revient à donner plus (resp. moins) d'importance à la descente de gradient. La valeur de λ est initialement fixée (typiquement $\lambda = 10^{-3}$).

- Si l'incrément $\boldsymbol{\Delta}_{LM}$ obtenu par la résolution de (1.51) permet une diminution de l'erreur, alors $\boldsymbol{\Delta}_{LM}$ est accepté et λ est divisé par 10 avant la prochaine itération.
- Au contraire, si $\boldsymbol{\Delta}_{LM}$ entraîne une augmentation de l'erreur, alors λ est multiplié par 10, on met à jour $\mathtt{N}'$ et on résout à nouveau (1.51) jusqu'à obtenir un $\boldsymbol{\Delta}_{LM}$ qui fait décroître l'erreur.

Si le coefficient λ est grand, alors la matrice $\mathtt{N}'$ de l'équation (1.51) est proche d'une matrice diagonale et le pas $\boldsymbol{\Delta}_{LM}$ correspond presque à une descente de gradient. Par contre, si λ est petit, le système à résoudre (1.51) se rapproche de (1.49) et l'on obtient un pas proche de celui de Gauss-Newton. L'algorithme est adaptatif et contrôle automatiquement l'amortissement. De ce fait, il est capable d'alterner entre une lente descente de gradient loin du minimum local et une rapide convergence quadratique dans son voisinage.
Contrairement à [30, 75], nous comptons une itération à chaque résolution du système (1.51). Cela évite d'avoir des itérations très longues en temps de calcul avec plusieurs résolutions de systèmes linéaires. Dans ce contexte, une itération n'améliore pas forcément l'erreur.

1.5.6 Mise en oeuvre de l'ajustement de faisceaux

Rappelons que $\mathbf{X}$ est le vecteur des paramètres à estimer et qu'il contient les paramètres extrinsèques des caméras et les coordonnées des points 3D. L'étape principale d'une itération d'ajustement de faisceaux est le calcul du pas Δ_{LM} par résolution du système linéaire

$$\mathtt{N}'\boldsymbol{\Delta}_{LM} = \mathtt{J}^\top \boldsymbol{\epsilon} \tag{1.52}$$

où la matrice $\mathtt{N}'$ est égale à la matrice $\mathtt{J}^\top\mathtt{J}$ dont les coefficients diagonaux sont multipliés par une constante.
L'ajustement de faisceaux est une étape fondamentale en reconstruction 3D. Cependant, son implémentation est souvent considérée comme délicate car elle nécessite :

- le calcul des dérivées partielles $\dfrac{\partial f_m}{\partial \mathbf{X}_n}$ (détaillé en Annexe 1).
- l'exploitation de la structure creuse de la matrice Jacobienne $\mathtt{J}$.

En effet, l'utilisation de la matrice Jacobienne $\mathtt{J}$ implique le calcul des dérivées partielles. Cependant, tous les points ne sont pas observés dans toutes

```
/* Fonction : Raffiner le vecteur de paramètres X ∈ R^N
   afin de minimiser ||Y − f(X)||^2 où f est la fonction de
   projection f : R^N → R^M(M ≥ N). Y ∈ R^M est le vecteur
   de mesure et l'estimation initiale est X_0 ∈ R^N. i_max
   est le nombre maximum d'itérations autorisé et α la
   réduction limite de l'erreur acceptée entre deux
   itérations.                                              */
début
    λ := 10^−3; i := 0; stop := faux;
    X := X_0;
    N := J^⊤J; ε := Y − f(X); g := J^⊤ε;
    tant que (non stop) and (i < i_max) faire
        N' := N ; diag(N') := diag(N') ∗ (1 + λ);
        Résoudre N'.Δ = g;
        X_new := X + Δ;
        ε_new := Y − f(X_new);
        si (||ε_new|| < ||ε||) alors
            X := X_new;
            si (||ε_new|| < α||ε||) alors
                λ := λ/10;
                N := J^⊤J; ε := Y − f(X); g := J^⊤ε;
            sinon
                stop := vrai;
        sinon
            λ := λ ∗ 10;
        fin
        i := i + 1;
    fin
fin
```

Algorithme 1 : L'algorithme de LEVENBERG-MARQUARDT.

les images et la matrice J contient des coefficients nuls. Cela implique, une structure spéciale par blocs pour la matrice $\mathtt{J}^\top\mathtt{J}$ (voir Figure 1.11). Cette matrice est composée de trois matrices : U, V, et W tels que U et V sont diagonales par blocs [30] :

– U, matrice carrée (de taille $6N_c \times 6N_c$) contenant des blocs diagonaux 6×6 représentant les relations entre les mesures dans une image i et

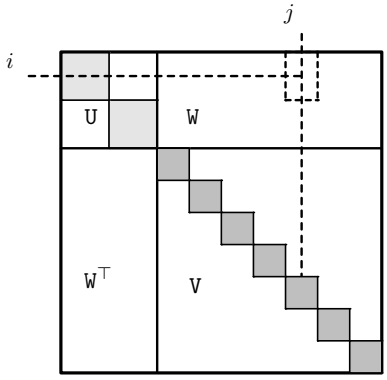


FIG. 1.11 – Structure de la matrice Hessienne approchée $\mathtt{J}^\top\mathtt{J}$.

les paramètres de caméra associés.
On peut expliciter la formule pour le bloc i :

$$\mathtt{U}^i = \sum_j \left(\frac{\partial f_j^i}{\partial C^i}(\mathbf{X})\right)^\top \left(\frac{\partial f_j^i}{\partial C^i}(\mathbf{X})\right) \tag{1.53}$$

où $\dfrac{\partial f_j^i}{\partial C^i}(\mathbf{X})$ est la matrice (de taille 2×6) contenant les dérivées partielles de f_j^i (projection du point j dans l'image i) par rapport aux paramètres de la caméra C^i.

– $\mathtt{V}$, matrice carrée (de taille $3N_p \times 3N_p$) diagonale par blocs 3×3 et donc facilement inversible. Chaque bloc représente les relations entre les paramètres d'un point j et les observations qu'on a de ce point.
La formule du bloc j est :

$$\mathtt{V}_j = \sum_i \left(\frac{\partial f_j^i}{\partial \mathbf{P}_j}(\mathbf{X})\right)^\top \left(\frac{\partial f_j^i}{\partial \mathbf{P}_j}(\mathbf{X})\right) \tag{1.54}$$

où $\dfrac{\partial f_j^i}{\partial \mathbf{P}_j}(\mathbf{X})$ est la matrice (de taille 2×3) contenant les dérivées partielles de f_j^i par rapport aux paramètres du point $\mathbf{P}_j$.

– $\mathtt{W}$, matrice (de taille $6N_c \times 3N_p$) exprimant les intercorrélations entre les paramètres des points 3D et les paramètres des caméras. La structure de $\mathtt{W}$ est directement liée au fait que les points sont vus ou non dans les images. $\mathtt{W}$ a un nombre de blocs 6×3 non nuls égal au nombre $N_r = M/2$ de reprojections 2D.

Le bloc $\mathtt{W}_j^i$ correspondant à f_j^i s'écrit :

$$\mathtt{W}_j^i = \left(\frac{\partial f_j^i}{\partial C^i}(\mathbf{X})\right)^\top \left(\frac{\partial f_j^i}{\partial \mathbf{P}_j}(\mathbf{X})\right). \tag{1.55}$$

Le système (1.51) se décompose alors :

$$\begin{bmatrix} \mathtt{U} & \mathtt{W} \\ \mathtt{W}^\top & \mathtt{V} \end{bmatrix} \begin{bmatrix} \mathbf{\Delta}_{cameras} \\ \mathbf{\Delta}_{points} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{cameras} \\ \mathbf{g}_{points} \end{bmatrix} \tag{1.56}$$

et peut être résolu en deux étapes [30] :

1. Le calcul du pas $\mathbf{\Delta}_{cameras}$ qui doit être appliqué aux paramètres des caméras par résolution du système linéaire suivant :
$$(\mathtt{U} - \mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top)\mathbf{\Delta}_{cameras} = \mathbf{g}_{cameras} - \mathtt{W}\mathtt{V}^{-1}\mathbf{g}_{points} \tag{1.57}$$
2. Le calcul direct du pas $\mathbf{\Delta}_{points}$ applicable aux points 3D :
$$\mathbf{\Delta}_{points} = \mathtt{V}^{-1}(\mathbf{g}_{points} - \mathtt{W}^\top \mathbf{\Delta}_{cameras}) \tag{1.58}$$

Étude sur la complexité d'une itération

Rappelons que N_p et N_c sont respectivement le nombre de points et de caméras optimisés dans l'ajustement de faisceaux. Soit p le nombre (considéré constant) de points se projetant dans chaque image. Le nombre de reprojections N_r est donc égal à $p.N_c$

Il faut tout d'abord construire la matrice $\mathtt{J}^\top\mathtt{J}$ (Figure 1.11). Cette étape demande un nombre d'opérations élémentaires proportionnel au nombre N_r de reprojections 2D prises en compte. Ensuite, les deux étapes consommatrices en temps de calcul sont :

- le calcul du produit matriciel $\mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top$
- la résolution du système linéaire pour les caméras (1.57).

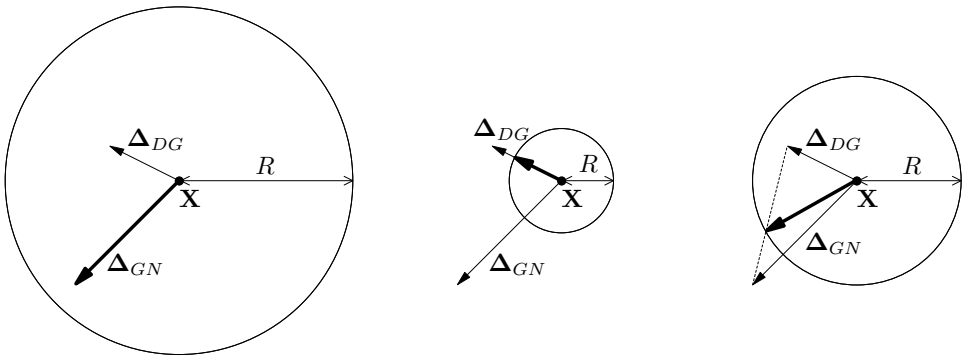
Pour le produit de matrices $\mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top$, le nombre d'opérations nécessaires peut être déterminé en considérant premièrement le nombre de blocs non-nuls de $\mathtt{W}\mathtt{V}^{-1}$. Il est égal au nombre de blocs de $\mathtt{W}$, c'est-à-dire $(p.N_c)$, nombre de reprojections dans N_c images, puisque $\mathtt{V}^{-1}$ est diagonale par blocs. Ensuite, dans le produit $(\mathtt{W}\mathtt{V}^{-1})\mathtt{W}^\top$, chaque bloc 6×3 non-nul de $\mathtt{W}\mathtt{V}^{-1}$ est utilisé une fois dans le calcul de chaque bloc colonne de $\mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top$. Ainsi, la complexité du produit $\mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top$ est $\Theta(p.N_c^2)$. La complexité de la résolution du système linéaire (1.57) est $\Theta(N_c^3)$ [75].
Finalement la complexité d'une itération d'ajustement de faisceaux est :

$$\Theta(N_r + p.N_c^2 + N_c^3). \tag{1.59}$$

1.5.7 L'algorithme "Dog Leg" de Powell

L'algorithme "Dog Leg" de Powell [74, 53, 49] est assez proche de celui de Levenberg-Marquardt car il combine Gauss-Newton et descente de gradient. Dans l'algorithme de Levenberg-Marquardt, il arrive que le coefficient λ soit mal choisi, ce qui peut entraîner des résolutions inutiles du système linéaire (1.51). Il arrive également que loin de la solution, on résout le même système alors que seul le gradient compte vraiment. Dans l'algorithme "Dog Leg" on cherche à déterminer le pas $\mathbf{\Delta}_{DL}$ en fonction de la confiance que l'on a dans le modèle quadratique de l'erreur au point $\mathbf{X}$. Powell a montré que $\|\mathbf{\Delta}_{DG}\| < \|\mathbf{\Delta}_{GN}\|$ et a proposé une façon élégante de paramétriser le choix de $\mathbf{\Delta}_{DL} = f_P(\mathbf{\Delta}_{GN}, \mathbf{\Delta}_{DG})$ en utilisant une boule de confiance de rayon R (illustré sur la Figure 1.12).

$$\mathbf{\Delta}_{DL} = \begin{cases} \mathbf{\Delta}_{GN} & \text{si } ||\mathbf{\Delta}_{GN}|| \leq R \\ \left(\frac{R}{\|\mathbf{\Delta}_{DG}\|}\right)\mathbf{\Delta}_{DG} & \text{si } ||\mathbf{\Delta}_{DG}|| \geq R \\ (1-\beta)\mathbf{\Delta}_{DG} + \beta\mathbf{\Delta}_{GN} & \text{sinon, avec } \beta \text{ tel que } ||\mathbf{\Delta}_{DL}|| = R \end{cases} \tag{1.60}$$



$||\mathbf{\Delta}_{GN}|| \leq R$ $\quad$ $||\mathbf{\Delta}_{DG}|| \geq R$ $\quad$ $||\mathbf{\Delta}_{DG}|| < R < ||\mathbf{\Delta}_{GN}||$

Fig. 1.12 – Région de confiance et pas "Dog leg" $\mathbf{\Delta}_{DL}$ (en gras).

On utilise le facteur de gain ρ pour contrôler le rayon de la boule de confiance.

$$\rho = \frac{||\mathbf{Y} - f(\mathbf{X})||^2 - ||\mathbf{Y} - f(\mathbf{X} + \mathbf{\Delta}_{DL})||^2}{\mathbf{E}(0) - \mathbf{E}(\mathbf{\Delta}_{DL})} \tag{1.61}$$

où $\mathbf{E}$ est l'approximation quadratique de l'erreur au point $\mathbf{X}$ pour un pas $\boldsymbol{\Delta}$ et s'écrit :

$$\mathbf{E}(\boldsymbol{\Delta}) = ||\boldsymbol{\epsilon} - \mathtt{J}\boldsymbol{\Delta}||^2 = \boldsymbol{\epsilon}^\top\boldsymbol{\epsilon} - 2(\mathtt{J}^\top\boldsymbol{\epsilon})^\top\boldsymbol{\Delta} + \boldsymbol{\Delta}^\top\mathtt{J}^\top\mathtt{J}\boldsymbol{\Delta}. \tag{1.62}$$

Si ρ est grand, cela indique que le modèle linéaire est bon, que l'on peut augmenter R et prendre un pas plus grand, proche de celui de GAUSS-NEWTON. Si ρ est petit (même négatif), alors il faut réduire R et prendre un pas plus petit, plus proche de la direction de descente de gradient.

Cette méthode évite de résoudre inutilement des systèmes linéaires, contrairement à LEVENBERG-MARQUARDT. Si on compte une itération à chaque résolution, alors le nombre d'itérations sera plus petit. Cependant, si le rayon de la boule de confiance est mal choisi et doit être ajusté, alors une itération peut prendre beaucoup plus de temps.

Comparaison avec Levenberg-Marquardt

Le Tableau 1.1 montre la comparaison entre la méthode de LEVENBERG-MARQUARDT et la méthode "Dog Leg". Deux séries d'ajustement de faisceaux ont été réalisées : une série de 5 longues séquences contenant chacune 30 images, et une série de 5 petites séquences de 3 images. Le tableau rapporte le nombre de points 3D estimés, l'erreur RMS initiale (en pixels) avant l'optimisation et l'erreur finale obtenue par les deux méthodes. On a également noté le nombre d'itération (nombre de résolution du système linéaire) et les temps de calcul (en secondes). On remarque que les résultats sont très similaires quelle que soit la méthode utilisée, tant au niveau de l'erreur qu'au niveau du temps de calcul. Comme on peut s'y attendre le nombre d'itérations est inférieur avec la méthode "Dog Leg" qu'avec la méthode de LEVENBERG-MARQUARDT. Cependant le temps moyen de chaque itération est supérieur (0.24 s contre 0.18 s pour la première série) et l'amélioration de l'erreur n'est pas meilleure. Finalement, les deux méthodes sont équivalentes pour les exemples testés. Par la suite, la méthode de LEVENBERG-MARQUARDT sera utilisée pour l'ajustement de faisceaux. Dans [49], LOURAKIS et ARGYROS ont obtenu de meilleurs résultats avec "Dog Leg" qu'avec LEVENBERG-MARQUARDT, en utilisant pour le second une méthode d'amortissement différente et vraisemblablement moins efficace. Avec notre augmentation de λ identique à [75, 30], la comparaison est beaucoup moins évidente.

/* **Fonction :** Raffiner le vecteur de paramètres $\mathbf{X} \in \mathbb{R}^N$ afin de minimiser $||\mathbf{Y} - f(\mathbf{X})||^2$ où f est la fonction de projection $f : \mathbb{R}^N \rightarrow \mathbb{R}^M (M \geq N)$. $\mathbf{Y} \in \mathbb{R}^M$ est le vecteur de mesure et l'estimation initiale est $\mathbf{X}_0 \in \mathbb{R}^N$. i_{max} est le nombre maximum d'itérations autorisé et α la réduction limite de l'erreur acceptée entre deux itérations. */

début

$R := R_0$; $i := 0$; $stop := faux$;

$\mathbf{X} := \mathbf{X}_0$;

$\mathtt{N} := \mathtt{J}^\top \mathtt{J}$; $\boldsymbol{\epsilon} := \mathbf{Y} - f(\mathbf{X})$; $\mathbf{g} := \mathtt{J}^\top \boldsymbol{\epsilon}$;

tant que (*non stop*) *and* $(i < i_{max})$ **faire**

$\boldsymbol{\Delta}_{DG} := \frac{||\mathbf{g}||^2}{||\mathtt{J}\mathbf{g}||^2}\mathbf{g}$;

Résoudre $\mathtt{N}.\boldsymbol{\Delta}_{GN} = \mathbf{g}$;

répéter

$\boldsymbol{\Delta}_{DL} := f_P(\boldsymbol{\Delta}_{DG}, \boldsymbol{\Delta}_{GN}, R)$ /* voir (1.60) */

$\mathbf{X}_{new} := \mathbf{X} + \boldsymbol{\Delta}_{DL}$;

$\rho := \frac{||\boldsymbol{\epsilon}||^2 - ||\mathbf{Y} - f(\mathbf{X}_{new})||^2}{E(0) - E(\boldsymbol{\Delta}_{DL})}$;

si $(\rho > 0)$ **alors**

$\mathbf{X} := \mathbf{X}_{new}$;

si $(||\boldsymbol{\epsilon}_{new}|| < \alpha||\boldsymbol{\epsilon}||)$ **alors**

$\mathtt{N} := \mathtt{J}^\top \mathtt{J}$; $\boldsymbol{\epsilon} := \mathbf{Y} - f(\mathbf{X})$; $\mathbf{g} := \mathtt{J}^\top \boldsymbol{\epsilon}$;

sinon

$stop := vrai$;

fin

/* mise à jour de R tirée de [53] */

si $(\rho > 0.75)$ **alors**

$R := max(R, 3 * ||\boldsymbol{\Delta}_{DL}||)$;

sinon si $(\rho < 0.25)$ **alors**

$R := R/2$;

fin

jusqu'à $(\rho > 0)$;

$i := i + 1$;

fin

fin

Algorithme 2 : L'algorithme "Dog Leg".

#img	#points	RMS initial	RMS Final		#itérations		temps(s)	
			DL	LM	DL	LM	DL	LM
30	1889	0.758	0.6596	0.6596	4	8	1.13	1.47
30	1542	0.722	0.5904	0.5904	4	4	1.10	0.90
30	1061	0.727	0.5703	0.5707	8	12	1.70	2.12
30	1312	0.881	0.7261	0.7262	3	9	0.73	1.78
30	1570	1.030	0.7719	0.7711	17	19	3.87	3.20
3	264	0.833	0.6799	0.6799	4	4	0.06	0.05
3	250	0.829	0.7108	0.7108	3	4	0.06	0.05
3	319	0.935	0.7958	0.7958	3	3	0.05	0.06
3	283	0.876	0.7468	0.7468	6	8	0.13	0.13
3	279	0.831	0.7241	0.7241	4	4	0.06	0.06

TAB. 1.1 – Comparaison entre "Dog Leg" (DL) et LEVENBERG-MARQUARDT (LM).

Chapitre 2

Systèmes de reconstruction 3D automatique existants

Nous venons de définir les outils élémentaires pour la reconstruction 3D à partir d'images multiples. Il existe de nombreuses façons de les combiner selon l'application désirée. Nous allons maintenant présenter les grandes familles de méthodes publiées concernant l'estimation du mouvement d'une caméra et la reconstruction 3D de l'environnement dans lequel elle évolue.

2.1 Les algorithmes de "SLAM"

De nombreux travaux sont regroupés sous le terme SLAM (Simultaneous Localization And Mapping) qu'on peut traduire par cartographie et localisation simultanées. Le SLAM a été initialement introduit à la fin des années 1980 par Leonard et Durrant-Whyte [40] en se basant sur les travaux de Smith et Cheeseman [94]. Le problème du SLAM est avant tout un problème de robotique : le but est de construire une carte d'un environnement inconnu dans lequel évolue un robot mobile. Au fur et à mesure de son déplacement, le robot se localise à partir de la carte et met à jour celle-ci afin de la rendre plus complète et plus précise. Le problème semble simple au premier abord, mais il est rendu complexe par la prise en compte des incertitudes inhérentes aux capteurs permettant d'accéder à la localisation du robot. A un instant donné, si la localisation du robot n'est pas précise, les nouveaux éléments ajoutés à la carte vont contenir les erreurs correspondantes. Si l'on ne tient pas compte de cela, alors les erreurs s'accumulent et dégradent la

qualité de la carte et de la localisation du robot. Le principe est très général et peut être employé avec de nombreux capteurs (télémètre laser, radar, caméra, odomètre, etc...) qui peuvent être combinés ou non entre eux. Par exemple, SE *et al.* [89] ont réalisé un couplage entre un odomètre et un système de vision trinoculaire. D'autre part, KIM et SUKKARIEH [37] ont utilisé la vision monoculaire combinée avec un capteur inertiel pour la localisation aérienne et la cartographie de terrain. Enfin, les caméras panoramiques, qui offrent un large champ de vue permettant un long suivi des amers, ont été récemment impliquées dans les systèmes de SLAM [39, 36].

Dans le cas où l'unique capteur de localisation est une caméra (ou un ensemble de caméras), on parle alors de SLAM par Vision ou *Vision-SLAM*. C'est un cas très particulier puisque les images fournies nécessitent des traitements postérieurs pour détecter et apparier les primitives. De plus, contrairement aux autres capteurs, au moins deux observations d'un même point sont nécessaires pour connaître sa position dans l'espace. La prise en compte de cette particularité a été largement étudiée et porte le nom de Bearing-only SLAM. En 2003, DAVISON [12] a proposé une méthode de SLAM par vision monoculaire en temps-réel (30 $images/s$). Son approche, tout comme la plupart des méthodes de SLAM, est basée sur l'utilisation d'un filtre de KALMAN étendu. Dans ce type d'approche, le vecteur d'état contient à la fois la pose courante de la caméra et la position de l'ensemble des points présents dans la carte. Une matrice de covariance est associée au vecteur d'état : elle permet de quantifier la confiance qu'on a sur la position de la caméra et des points. Le principe est le suivant : il s'agit d'estimer conjointement et incrémentalement la position du robot et la carte des amers. A chaque instant, le vecteur d'état est estimé en fusionnant un modèle prédit qui dépend des évènements passés et un modèle d'observation calculé à partir des mesures fournies par les capteurs. Ainsi, les points grossièrement estimés au départ peuvent être affinés en cohérence avec les nouvelles observations. L'avantage de cet algorithme est la possibilité de retrouver des points alors qu'ils n'étaient plus dans le champ de vue car ceux-ci restent dans le vecteur d'état. En revanche, la principale limite est l'accroissement du temps de calcul de la mise à jour du filtre de KALMAN lorsque la taille de la carte augmente. La complexité de la mise à jour est en $\Theta(N^2)$, où N est la taille du vecteur d'état. Cela signifie que faire du SLAM temps-réel avec cette technique n'est possible que dans un environnement de taille réduite où le nombre d'amers utilisés est limité. Pour aller au delà, il est alors nécessaire de bien gérer la carte comme le font CHIUSO *et al.* dans une approche très similaire [8] en ajoutant ou en supprimant des points. Il est également possible d'utiliser des sous-cartes

en mettant à jour une carte locale de l'environnement immédiat du robot et en changeant de sous-carte lorsque le robot en sort. Il faut alors gérer la cohérence entre les sous-cartes comme le proposent Newman et Leonard [67].
Une tendance récente consiste à remplacer le filtre de Kalman par un filtre à particules dans les systèmes de SLAM. L'algorithme général, publié par Montermerlo *et al.* [61], a été baptisé *FastSLAM*. La complexité de calcul obtenue est en $\Theta(M \log(N))$, où M est le nombre de particules. La fonction de distribution de la pose du robot est représentée par un ensemble de particules, chacune d'elles encodant une trajectoire complète du robot et contenant une carte. Chaque carte est alors composée d'un ensemble d'amers qui sont tous estimés par un simple filtre de Kalman à petite dimension. L'intérêt de cette technique repose sur le fait que le problème de SLAM est partitionné en un problème de localisation et un problème de cartographie. L'approche *FastSLAM* a ensuite été appliquée au SLAM par Vision [92, 97].

2.2 Les algorithmes de "SfM"

Le terme *Vision-SLAM* utilisé par la communauté de robotique, désigne sensiblement le même problème que l'expression "Structure from Motion" (SfM) plus courante dans la communauté de la vision par ordinateur. La problématique de traiter automatiquement une séquence vidéo pour obtenir une reconstruction 3D se retrouve également dans les travaux de vision. La différence majeure entre les deux approches est que la reconstruction en SLAM se fait toujours de manière incrémentale alors que les algorithmes de vision nécessitent parfois de disposer de l'ensemble des images de la séquence vidéo à traiter. Selon la communauté, les outils utilisés pour traiter le problème sont souvent différents : outils statistiques pour le SLAM (filtre de Kalman, filtre à particules, etc...) et outils géométriques en vision (méthodes robustes, ajustement de faisceaux). Il est reconnu que la méthode de reconstruction 3D la plus précise est l'ajustement de faisceaux [13, 22, 102] mais c'est également la méthode qui a la plus grande complexité de calcul. En effet, le principal inconvénient de l'ajustement de faisceaux est qu'il demande des temps de traitement très longs, ce qui l'empêche d'être utilisé dans un contexte temps-réel ou sur de longues séquences vidéo. C'est pour cela que la plupart des méthodes rapides (souvent incrémentales) ne font pas intervenir l'ajustement de faisceaux.

2.2.1 Les algorithmes rapides sans ajustement de faisceaux

Les travaux de SfM temps-réel tendent à se rapprocher du SLAM en procédant de manière incrémentale mais sans intégrer les notions de prédiction, d'incertitude et d'identification des amers. Les algorithmes de vision fonctionnent souvent suivant ce principe : détection des primitives à reconstruire, mise en correspondances de ces primitives, et finalement calculs géométriques 3D. Les premières expériences temps-réel datent de 1995 avec les travaux de TOMASI *et al.* [100]. L'algorithme le plus abouti est celui de NISTER *et al.* [70] publié en 2004 sous le nom "Visual Odometry", c'est-à-dire l'estimation du mouvement de la caméra à partir des seules données visuelles. Le système opère en temps-réel dans un but de guidage pour la navigation, sans aucune connaissance *a priori* de l'environnement ou de la nature du mouvement. Le premier point du système est la détection d'un nombre important de points d'intérêt (jusqu'à 5.000 points de HARRIS par image). Les points détectés sont appariés entre les images et reliés à la vitesse de la vidéo ($\approx 13\ images/s$). La mise en correspondance est réalisée à partir d'un calcul de corrélation normalisée sur des imagettes 11×11. Cette étape est rendue très rapide par l'utilisation du jeu d'instructions MMX. La méthode de reconstruction présentée au chapitre 3 est basée sur une mise en correspondance d'images très similaire. L'initialisation est obtenue à partir des trois premières images et l'algorithme des 5 points [69] en conjonction avec RANSAC. Ensuite les points sont reconstruits en utilisant une simple triangulation et les poses de caméra sont calculés à l'aide d'un algorithme 3 points [24] et RANSAC. L'algorithme ne cherche pas à identifier des points qui ont déjà été cartographiés, puis qui sont sortis du champ de vision et qui redeviennent visibles. De plus il y a un grand risque de dérive à cause de l'accumulation des erreurs au fil du temps. Pour diminuer ces erreurs, une possibilité est d'utiliser l'ajustement de faisceaux.

2.2.2 Les algorithmes avec ajustement de faisceaux global

L'ajustement de faisceaux offre la reconstruction 3D la plus précise puisque les paramètres des points 3D et les poses des caméras sont optimisés pour minimiser l'erreur de reprojection globale. Cependant, il est impossible de calculer la reconstruction sur une séquence entière (par exemple avec la méthode précédente) et d'appliquer ensuite l'ajustement de faisceaux car l'estimation

initiale serait trop éloignée de la vraie solution. Cela peut entraîner des problèmes de convergence vers un mauvais minimum local. Pour remédier à cela, il faut procéder de manière séquentielle.
Tout d'abord, l'ajustement peut être géré de manière hiérarchique comme dans les travaux de Fitzgibbon et Zisserman [22] ou Royer *et al.* [83] : une longue séquence est découpée en sous-séquences de trois images. La géométrie est calculée pour chaque triplet, et ceux-ci sont optimisés et fusionnés deux à deux pour obtenir des sous-séquences plus grandes. Les sous-séquences sont ensuite elles-mêmes optimisées puis fusionnées, et ainsi de suite jusqu'à la séquence complète. L'étape finale est une optimisation globale de tous les paramètres 3D. Cette technique nécessite de disposer de l'ensemble des images avant de démarrer le processus de reconstruction.
La deuxième solution est de procéder incrémentalement, en traitant les images dans l'ordre de la séquence vidéo. Dans ce cas, le calcul peut débuter avant même la fin de l'acquisition des images mais il s'agit d'optimiser la structure complète à chaque fois que l'on ajoute des éléments 3D. Plusieurs auteurs [73, 68, 22] ont remarqué que procéder à un ajustement de faisceaux à chaque fois qu'on ajoute une image améliore les chances de réussite du processus. Les deux approches sont illustrées sur la Figure 2.1 où les étapes d'ajustement sont représentées par les ellipses numérotées.

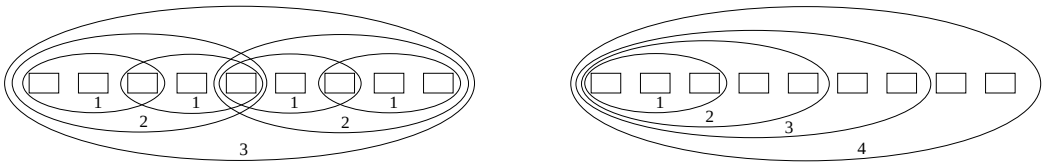


Fig. 2.1 – Deux stratégies pour l'ajustement de faisceaux global : hiérarchique (gauche) et incrémentale (droite).

Ces méthodes sont intéressantes pour traiter des séquences avec un nombre d'images limité et des structures simples (quelques centaines à quelques milliers de points dans [73]). L'inconvénient est que l'ajustement de faisceaux global devient extrêmement coûteux voire même impossible à réaliser pour de longues séquences. Différentes variantes ont naturellement été proposées par la communauté afin d'accélérer les calculs et de pouvoir traiter des quantités de données plus importantes.

2.2.3 Les algorithmes avec ajustement de faisceaux modifié

Les trois principales voies explorées afin de réduire la complexité de calcul de l'ajustement de faisceaux sont les suivantes :

1. La prise en compte de la structure éparse des données qui provient de l'indépendance des points et des caméras entre eux et du fait que tous les points ne sont pas visibles dans toutes les images.
2. L'optimisation alternée des points et des caméras.
3. La réduction du nombre de paramètres optimisés en rejetant des points ou des vues, ou en séparant les données en sous-séquences.

Tout d'abord, la première solution ou l'implémentation dite "creuse" est bien connue depuis les photogrammètres et est désormais courante dans les travaux de vision par ordinateur [17, 48, 30, 102].
La seconde solution qui consiste à alterner la minimisation de l'erreur de reprojection en faisant varier les points et ensuite les caméras est intéressante. Puisque chaque point est estimé indépendamment étant données les caméras fixes et réciproquement chaque caméra est estimée indépendamment à partir des points fixes, la plus grande matrice inversée est celle qui est utilisée pour le calcul d'une caméra. La solution obtenue est théoriquement la même qu'avec l'ajustement de faisceaux standard car c'est le même critère qui est minimisé. Cependant, la convergence peut être plus longue [102].
Contrairement aux deux premières méthodes, la troisième dégrade la qualité de la reconstruction puisque la solution n'est pas optimale pour l'ensemble des données. Cette voie a cependant été très explorée et la dégradation est souvent minime par rapport au temps de calcul gagné. Les deux stratégies de reconstruction (hiérarchique et incrémentale) ont été étudiées.

Dans [91], SHUM *et al.* présentent une approche hiérarchique efficace. Une longue séquence est divisée en plusieurs segments. Pour chaque segment, un modèle 3D optimal est calculé par ajustement de faisceaux. Ensuite, les modèles sont fusionnés dans un système de coordonnées commun et des images clef virtuelles sont extraites pour chaque segment. Chaque paire d'entre-elles encode les informations relatives à la structure 3D d'un segment et à son incertitude. Le modèle 3D de la séquence complète est obtenu avec un ajustement de faisceau appliqué uniquement sur les images clef virtuelles, ce qui diminue largement le nombre de paramètres impliqués.

L'approche incrémentale a davantage été investiguée [95, 108, 46, 17, 105].

Un algorithme incrémental d'estimation du mouvement de la caméra a été proposé par Zhang et Shan [108]. Il s'applique sur une fenêtre glissante de trois images et sur des points vus dans au moins deux images. Pour chaque image I_i, on détermine le mouvement M_i en appliquant l'ajustement local au triplet (I_{i-2}, I_{i-1}, I_i) à partir des points appariés sur deux ou trois vues (Figure 2.2). Deux implémentations sont décrites : la première (exacte) optimise les paramètres 3D de la structure et les paramètres de pose des caméras. La seconde simplifie le problème d'origine, la minimisation portant seulement sur les paramètres des caméras. Cette approche entraîne des problèmes de dérive car les triplets optimisés ne sont pas obligatoirement consistants entre eux.

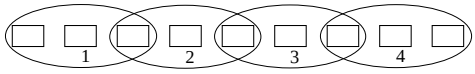


Fig. 2.2 – Ajustement de faisceaux incrémental et local.

Steedly et Essa [95] ont présenté une méthode de SfM incrémentale qui prend en compte l'information innovante apportée par les nouvelles images. Les nouvelles données sont ajoutées à une solution existante dont l'erreur a été minimisée au sens des moindres carrés. En traçant des connexions entre les points et les caméras, on détermine les paramètres qui ont besoin d'être ajustés et on évite ainsi d'optimiser la totalité des poses des caméras et de la structure de la scène. Une fois le graphe des dépendances réalisé, on commence par optimiser la pose de la nouvelle caméra. Ensuite, on regarde si les points qui y sont connectés sont "tirés" par rapport à leur position actuelle. Si c'est le cas, on les relâche pour qu'ils puissent bouger. Et ainsi de suite, on revient en arrière dans la structure jusqu'à trouver un état stable. Dans la plupart des cas, la complexité de l'ajustement lors de l'ajout d'une nouvelle caméra est beaucoup plus petite que celle d'un ajustement de faisceaux global. Ce principe a été repris très récemment par Zhang *et al.* [105] sous l'appellation "Local On-Demand Bundle Adjustment" qui signifie : ajustement de faisceaux local à la demande.

En 2006, Engels *et al.* [17] ont étudié l'influence que pouvait avoir un ajustement de faisceaux appliqué incrémentalement sur le processus de reconstruction 3D. Les auteurs ont remarqué que cela n'apporte pas seulement de la précision mais également de la robustesse. En effet, il est par exemple évident que le calcul de pose pour une nouvelle image a une plus grande probabilité d'être correct si les points utilisés dans ce calcul ont été affinés auparavant. Le principe développé est le suivant : à chaque fois qu'une nouvelle

image est ajoutée, n itérations d'ajustement de faisceaux sont appliquées sur une sous-séquence qui contient les m dernières images. Le critère minimisé inclut les erreurs de reprojections robustes ($\rho(\epsilon) = \ln\left(1 + \frac{\epsilon^2}{\sigma^2}\right)$) des points dans toutes les images, y compris dans les images des caméras fixes. L'étude de l'influence du couple de paramètres (m, n) sur la vitesse de calcul et sur le taux de réussite du processus indique que l'ajustement de faisceaux peut désormais faire partie intégrante d'un système de reconstruction temps-réel. Ce travail a été réalisé en même temps que la thèse et la méthode décrite ici est très proche de celle présentée au chapitre 3.

Les algorithmes développés dans [13, 57] combinent les concepts d'estimation récursive par filtre de KALMAN et d'optimisation par ajustement de faisceaux, en partant du constat que l'optimisation globale apporte une meilleure précision mais nécessite une plus grande complexité de calcul. L'approche combinée fournit des résultats comparables aux moindres carrés traditionnels mais avec une complexité proche de celle d'un filtre de KALMAN.

2.3 Discussion et positionnement

Pour résumer, les deux approches (SLAM et SfM) se distinguent notamment par leur manière différente d'intégrer les nouvelles observations à l'instant t dans une reconstruction existante. Ces observations contiennent des informations qui permettent le calcul de paramètres nouveaux (la pose de la caméra qui a changé) ou le renforcement de plus anciens (les points visibles déjà reconstruits). Dans tous les cas, les nouvelles observations doivent être compatibles et cohérentes avec les paramètres déjà calculés. Dans la première approche (SLAM), cette compatibilité est assurée par le modèle et les covariances sur le modèle à l'instant $t-1$. En effet le vecteur d'état à l'instant t est calculé à partir de ces informations à l'instant $t-1$. On ne revient jamais au delà de $t-1$ (aspect purement récursif). Dans la thèse, on présente une méthode complète de reconstruction 3D incrémentale apparentée davantage aux algorithmes de SfM. Elle est basée sur la détection et la mise en correspondance de points d'intérêt très similaires à [70, 83] avec un affinement de la structure par ajustement de faisceaux local proche de [108, 17]. Dans ce type d'approche de SfM, les observations à l'instant t sont ajoutées aux précédentes $t-1$, $t-2$ (dans [108]) et même davantage $t-3$, $t-4$, etc... (dans [95, 17]). Le modèle peut alors être affiné à partir de toutes les observations, ce qui apporte une plus grande précision. Les auteurs de [17] considèrent même que leur système (proche de celui développé ici) est équivalent à un

filtre de KALMAN si les paramètres n et m sont réduits à 1 et au moins aussi précis dans les autres cas. Le cas le plus précis est l'ajustement de faisceaux global où tous les paramètres sont optimisés en tenant compte de toutes les observations. L'idée développée dans la thèse est de réduire le nombre de ces paramètres afin d'accélérer les traitements, tout en vérifiant que la perte de précision est minime.

Dans les systèmes de reconstruction automatique existants, on remarque que plusieurs problèmes sont couramment rencontrés. Un premier problème concerne la répétabilité de localisation, c'est-à-dire la capacité de déterminer si la caméra se situe dans une position qu'elle a déjà occupé auparavant (ou une position proche) après s'être déplacée. Pour cela, il s'agit souvent de retrouver des points enregistrés qui ont disparu du champ de vue et qui sont redevenus visibles. Si on ne les retrouve pas, alors on considère que ce sont des nouveaux points et ils peuvent être de nouveaux reconstruits. Ainsi, un même point de la scène peut être reconstruit en deux points différents (ce qui est fait dans nos travaux). Les algorithmes de SLAM gèrent bien ce cas à petite échelle (petits mouvements de caméra) puisque que l'on connaît la position de la caméra, la position des points et donc la prévision de leur position dans l'image. A plus grande échelle, le problème est beaucoup plus compliqué puisque la trajectoire de la caméra a dérivé.
En effet, la dérive est sans doute le problème majeur auquel on est confronté lorsqu'on traite des séquences d'images où la caméra parcourt une longue distance (plusieurs centaines de mètres). Celle-ci est essentiellement dûe à l'accumulation des erreurs dans les algorithmes récursifs. Le calibrage joue également un rôle important à cause de la "dérive projective" [10, 78]. Elle se traduit souvent par la modification du repère de la reconstruction au cours du temps : les éléments calculés en fin de reconstruction ne sont pas cohérents avec les premiers. Dans [10], la dérive est détectée et corrigée à partir des instances multiples des mêmes points de la scène et un ajustement adapté qui prend en compte des correspondances de points 3D-3D. La méthode développée dans la thèse, accompagnée d'un calibrage précis de la caméra, a pour but de produire une faible dérive.
Ce même problème intervient dans le cas particulier des trajectoires en boucles, qui sont très difficiles à traiter sur des longues séquences. Certains travaux proposent une détection automatique des boucles : par exemple par une mise en correspondance basée sur l'apparence d'une nouvelle image avec une base d'images enregistrées [39] ou par des similarités géométriques et photométriques des points d'intérêt reconstruits [9]. Une approche plus simple [22] consiste à imposer que la première et la dernière image de la séquence

vidéo correspondent au même point de vue. Ces informations particulières ne sont pas utilisées dans nos travaux.

Un problème se pose également lorsque la fréquence d'acquisition est grande par rapport à la vitesse de déplacement de la caméra. Dans ce cas, il est inutile de traiter l'ensemble des images puisque l'information nouvelle apportée par chaque image est faible. Il semble donc tout naturel de sélectionner un sous ensemble d'images pour procéder aux calculs. Il n'existe pas de méthode universelle pour cela mais les auteurs de [99, 78] sont d'accord pour dire que la complexité de la tâche réside dans le compromis entre des images éloignées facilitant les calculs géométriques (grande "baseline") et des images proches assurant une meilleure mise en correspondance. Dans la thèse, on utilise le terme d'"images clef" et on présente une méthode simple pour le choix de ces images particulières.

Finalement, le problème du temps de calcul reste à ce jour encore d'actualité et peu de systèmes rapides existent pour traiter des longues trajectoires de manière précise.

Chapitre 3

Reconstruction 3D incrémentale par ajustement de faisceaux local : cas d'une caméra perspective

Dans ce chapitre, on présente en détails la méthode de reconstruction 3D réalisée au cours de la thèse. C'est un algorithme de reconstruction incrémental, basé sur la mise en correspondance d'images à l'aide de points d'intérêt. Ces points d'intérêt sont reconstruits conjointement avec les poses de la caméra tout au long de la séquence vidéo. La contribution majeure est l'introduction d'une méthode d'optimisation par ajustement de faisceaux local et ces travaux ont donné lieu à plusieurs publications [63, 65, 64].

3.1 Aperçu de l'approche

Considérons une séquence vidéo acquise par une caméra se déplaçant dans un environnement inconnu. Le but de ce travail est de déterminer la position de la caméra et son orientation dans un repère global à différents instants t, ainsi que la position d'un ensemble de points 3D observés au long de la séquence. On dispose pour cela d'une caméra monoculaire perspective dont les paramètres intrinsèques (y compris les paramètres de distorsion radiale) sont connus et supposés inchangés durant la séquence. L'algorithme développé s'appuie sur la mise en correspondance d'images à partir de la détection de

points d'intérêt appariés par un calcul de corrélation (section 3.2). Une étape d'initialisation (section 3.3) consiste à choisir un premier triplet d'images afin de définir le repère global et calculer la géométrie relative. Ensuite, à partir des correspondances de points, on procède au calcul robuste de la pose de la caméra pour chaque image du flux vidéo (section 3.4.1). On effectue alors un sous-échantillonnage temporel et certaines images sont sélectionnées automatiquement en tant qu' "images clef" pour la triangulation des points 3D (section 3.4.2). La méthode opère de manière "incrémentale" et, lorsqu'une nouvelle image clef est ajoutée, on reconstruit de nouveaux points 3D et on réalise un ajustement de faisceaux appliqué localement sur la fin de la séquence (section 3.5).

Le résultat (voir Figure 3.1) est un ensemble de poses représentant la trajectoire de la caméra, ainsi que les coordonnées 3D des points observés dans les images.

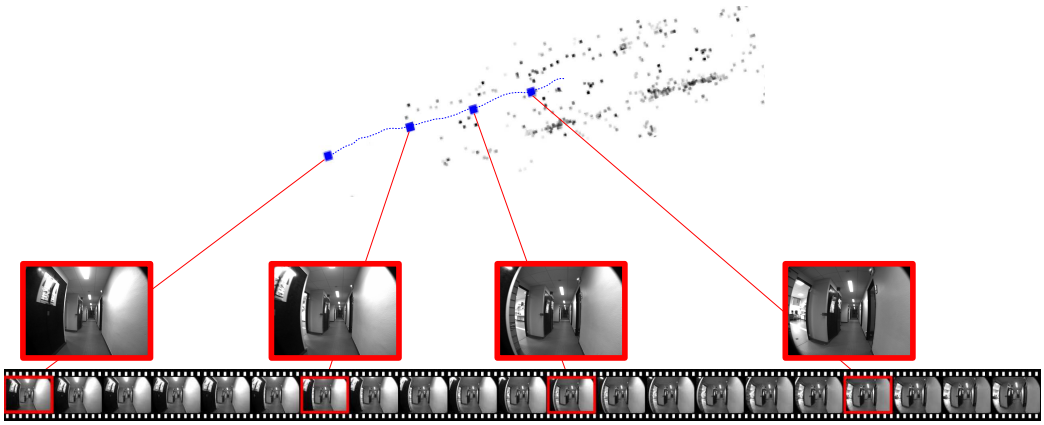


FIG. 3.1 – Sous-échantillonnage temporel de la séquence vidéo et vue de dessus de la reconstruction 3D : localisation de la caméra pour chaque image (trait continu), reconstruction des points 3D (points) à partir des images clef (carrés).

3.2 Mise en correspondance d'images : choix techniques

3.2.1 Détection de points d'intérêt

La méthode retenue est basée sur le calcul du critère de HARRIS $C_H(x, y)$ pour chaque pixel (x, y). Les points où le critère est à un maximum local sont susceptibles d'être choisis comme points d'intérêt, autrement dit un pixel (x, y) peut devenir un point d'intérêt seulement si $C_H(x, y) > C_H(x+i, y+j)\ \forall i, j \in \{-1, 0, 1\}$. Cependant, on ne peut pas choisir tous les maxima locaux comme points d'intérêt car ils sont trop nombreux pour pouvoir envisager ensuite une mise en correspondance rapide. D'abord, on choisit les N maxima dans l'image où le critère de HARRIS est le plus fort. Ensuite, afin de garantir une bonne répartition des points d'intérêt dans l'image, celle-ci est découpée en $8 \times 8 = 64$ baquets. On ajoute alors les M meilleurs points de chaque baquet. Habituellement, on choisit $N = 500$ et $M = 20$ pour une taille d'image de l'ordre de 512×384.

Les points sont ainsi détectés au pixel près et les coordonnées (x, y) obtenues sont des valeurs entières. Une deuxième étape consiste à affiner de manière subpixellique (plus précise que le pixel) la position de chaque point d'intérêt sélectionné. La recherche de la position précise du maximum du critère est réalisée indépendamment en x et en y. Pour déterminer l'abscisse, on recherche le maximum d'un polynôme $P(x)$ de degré 2 qui interpole le signal tel que $P(x-1) = C_H(x-1, y)$, $P(x) = C_H(x, y)$, et $P(x+1) = C_H(x+1, y)$. L'ordonnée du maximum est obtenue de la même façon. Si le point (x, y) est un maximum local, alors le point $(\tilde{x}, \tilde{y})$ subpixellique se trouve à l'intérieur de l'imagette délimitée par les points $(x-1, y-1)$, $(x+1, y-1)$, $(x-1, y+1)$ et $(x+1, y+1)$.

3.2.2 Appariements des points

On dispose maintenant de deux images I^1 et I^2 dont on a extrait les points d'intérêt. Le but est de faire correspondre les points de la première image aux points de la deuxième image. On doit donc pour cela créer une liste de paires de points homologues $(\mathbf{p^1}, \mathbf{p}^2)$. Le critère retenu pour affirmer si un couple de points $(\mathbf{p^1}, \mathbf{p}^2)$ forme une paire de points homologues est le calcul d'un score de corrélation ZNCC (Zero Normalized Cross Correlation) sur le voisinage (imagette de taille 11×11) des points $\mathbf{p^1}$ et $\mathbf{p}^2$. Ce score normalisé se rapproche de 1 si les régions sont semblables et diminue dans le

cas contraire.

La procédure se déroule de la façon suivante : on parcourt tous les points d'intérêt de l'image I^1 et pour chaque point $\mathbf{p}^1$, on définit une zone de recherche (ou Region Of Interest, ROI) rectangulaire centrée en les coordonnées de $\mathbf{p}^1$ dans l'image I^2. On calcule le score ZNCC obtenu avec tous les points de l'image I^2 situés dans la zone de recherche pour ne retenir que le score le plus fort, à condition que celui dépasse un seuil fixe (par exemple 0.8). Le réglage de la taille de fenêtre (11×11) et du seuil ZNCC (0.8) est le même que dans [43]. Pour satisfaire la contrainte d'unicité (un point de l'image I^2 ne peut pas être apparié à deux points différents de l'image I^1), on vérifie que tous les points de l'image I^2 ne sont choisis qu'une seule fois. Si ce n'est pas le cas, alors on garde le couple qui a donné le meilleur score de corrélation ZNCC.

3.2.3 Mise en correspondance de l'image courante avec la dernière image clef

Afin de calculer la pose $C^i = (\mathtt{R}^i, \mathbf{t}^i)$ de la caméra correspondante à l'image courante i, il est nécessaire de mettre en correspondance les points 2D visibles dans l'image courante avec des points préalablement reconstruits en 3 dimensions dans le repère "monde". Les points étant reconstruits en 3D avec les images clef, chaque image clef contient des correspondances 3D/2D. En appariant les points de l'image courante avec les points de la dernière image clef, on obtient les correspondances 3D/2D nécessaires au calcul de la pose courante C^i. Pour cela, deux méthodes ont été développées (voir Figure 3.2) : une première méthode qui consiste à faire une mise en correspondance directe de l'image courante avec la dernière image clef sans se préoccuper des images intermédiaires. La deuxième méthode est une méthode dite "de proche en proche" où les appariements sont faits entre images successives depuis la dernière image clef jusqu'à l'image courante.

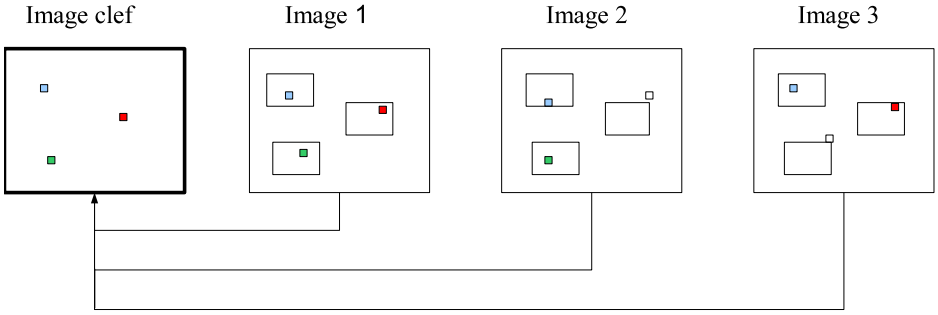
Mise en correspondance directe

Pour chaque point d'intérêt détecté dans la dernière image clef, on définit une zone de recherche fixe (typiquement de taille 40×30 pixels pour des images 512×384), centrée autour du point. On recherche alors, dans les images courantes suivantes, les points d'intérêt détectés dans cette zone, susceptibles de correspondre à ce point, pour ne retenir que celui qui donne un meilleur score de corrélation. Cette zone reste fixe selon les images, et l'on se

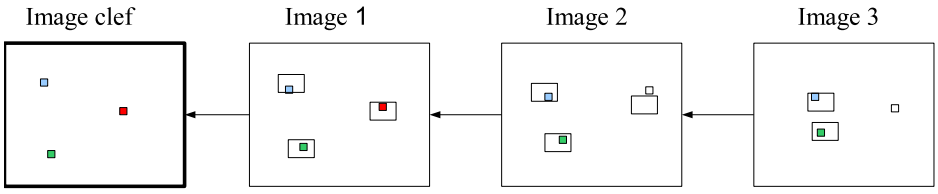
rend bien compte que plus la caméra s'éloigne de la position qu'elle occupait au choix de l'image clef, moins le nombre de points en correspondance sera grand. Si la caméra se rapproche, alors ce nombre de points augmente.

Mise en correspondance "de proche en proche"

Dans cette méthode, on exploite le fait que le mouvement relatif des points dans l'image est assez faible entre deux images successives. Pour la première image qui suit l'image clef, on utilise des zones de recherche réduites (10×8 pixels), car l'image est très proche de l'image clef. Une fois la mise en correspondance réalisée, on ne garde que les points de l'image qui ont été appariés avec un point de l'image clef et on met à jour les zones de recherche réduites autour de ces points dans l'image suivante, et ainsi de suite. Cette méthode permet d'accélérer les calculs car, le nombre et la taille des fenêtres de recherche sont réduits. Par contre, le nombre de correspondances ne fait que diminuer au long de la séquence, y compris si la caméra se rapproche de sa position initiale.



(a) directe.



(b) de proche en proche.

FIG. 3.2 – 2 méthodes pour la mise en correspondance : (a) directe. (b) de proche en proche (les points correctement appariés avec ceux de l'image clef sont en couleur).

Comparaison des 2 méthodes

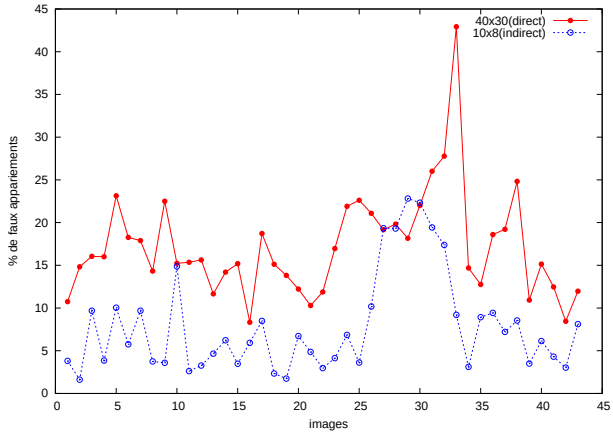
De manière intuitive, la méthode indirecte est intéressante : elle permet de filtrer les points ayant un grand déplacement entre deux images successives tout en autorisant des forts déplacements entre les images clef puisque les centres des zones de recherche se déplacent. En réalité, la taille des fenêtres ROI fixe l'amplitude du déplacement relatif maximum d'un point entre deux images successives alors que dans la méthode directe, elle limite ce même déplacement entre deux images clef.

Pour comparer les deux méthodes, on a utilisé une séquence d'images réelles dont on a préalablement calculé la géométrie de manière précise et robuste. On connaît alors la pose de la caméra pour plusieurs images (images clef) parmi la série d'images complète. La géométrie épipolaire est donc connue pour deux images clef successives, et on peut ainsi vérifier pour chaque couple de points pris dans ces deux images si la contrainte épipolaire est respectée. En réalité, on mesure la distance d'un des deux points par rapport à la droite épipolaire calculée à partir de l'autre point. Si la distance est supérieure à un seuil fixé (ici 5 pixels), alors on considère que l'on a affaire à un faux appariement et les points ne sont pas deux points homologues. Si la distance est inférieure au seuil, alors il est plus probable que l'appariement soit correct. Les résultats des tests sont présentés sur la Figure 3.3. Le graphique 3.3(a) montre le pourcentage de faux appariements obtenus avec chacune des deux méthodes au long de la séquence. Les résultats sont présentés avec des tailles de fenêtre de recherche 40×30 pixels pour la méthode directe et 10×8 pour la méthode de proche en proche. On voit que la méthode directe offre un pourcentage de faux appariements beaucoup plus important que la méthode indirecte. Avec le Tableau 3.3(b) de la Figure 3.3, on remarque également que le taux d'erreur et le temps de calcul diminuent lorsqu'on prend des zones de recherche plus petites. Cependant, prendre des fenêtres trop petites est risqué car on élimine les points qui ont un mouvement relatif important dans les images.

Le tableau montre aussi le nombre total d'appariements réalisés avec les deux méthodes ainsi que le nombre d'appariements considérés comme corrects avec la géométrie épipolaire. La méthode directe offre un nombre d'appariements corrects plus importants, mais le nombre global de couples calculé est aussi beaucoup plus grand.

Afin de limiter la proportion de mauvais appariements et ainsi le nombre de tirages dans les algorithmes robustes, on préférera utiliser la méthode indirecte qui offre un nombre de couples suffisant et un meilleur taux d'ap-

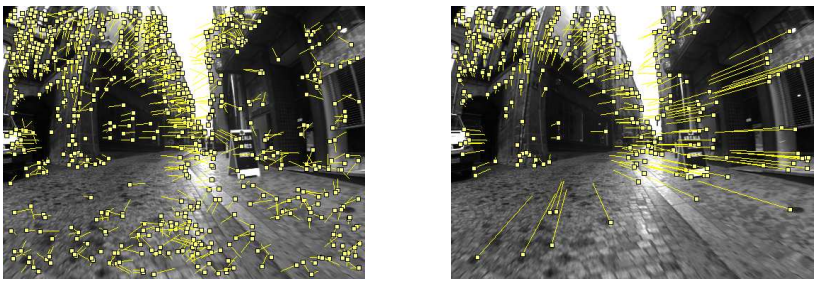
pariements corrects avec des temps de calcul inférieurs.



(a) Pourcentage de faux appariements.

Méthode	Taille ROI	#Couples total	#Corrects	%Corrects	Temps (ms)
directe	40×30	446.6	370.7	82.8%	78.2
directe	30×20	395.0	338.8	84.1%	61.0
indirecte	15×12	303.7	266.4	87.3%	55.9
indirecte	10×8	274.8	254.6	92.2%	52.9

(b) Nombre total de couples, nombre et pourcentage de couples vérifiant la géométrie épipolaire.



(c) Appariements entre images clef (gauche : méthode directe 40×30, droite : méthode indirecte 10×8).

FIG. 3.3 – Comparaison des 2 méthodes d'appariements : "directe" et de "proche en proche" (indirecte).

3.3 Initialisation robuste avec trois vues

3.3.1 Choix du premier triplet d'images clef

On garde à l'esprit que le mouvement de la caméra doit être suffisamment important pour calculer précisément la géométrie épipolaire tout en assurant une mise en correspondance fiable. Pour cette raison, les premières images clef sont sélectionnées comme dans [83], relativement éloignées les unes des autres mais avec un nombre important de points en correspondance. Pour cela, la première image I^0 est toujours choisie comme image clef. La deuxième image clef I^1 est sélectionnée la plus loin possible dans la séquence vidéo mais avec au moins M points appariés avec les points de l'image I^0. Ensuite, pour I^2, on prend l'image la plus éloignée qui a au moins M points appariés avec les points de I^1 et au moins M' points appariés avec ceux de I^0.
Par exemple, dans nos expériences on a choisi les valeurs $M = 400$ et $M' = 300$.

3.3.2 Initialisation de la géométrie

Ce procédé permet d'obtenir un nombre suffisant de points en correspondance pour calculer le mouvement relatif de la caméra. Le repère lié à la caméra 0 est choisi comme repère global de la reconstruction. Sa pose est donc $(\mathtt{R}^0, \mathbf{t}^0) = (\mathtt{I}_3, [0\ 0\ 0]^\top)$. On calcule le mouvement relatif $(\mathtt{R}, \mathbf{t})$ de la caméra entre la première et la troisième image, à partir de l'algorithme des 5 points [69] présenté dans la section 1.3.1 et une approche basée sur RANSAC [21]. L'utilisation de trois vues au lieu de deux améliore la robustesse et permet d'éliminer les ambiguïtés induites par les points coplanaires [69]. La méthode complète est expliquée de manière détaillée dans la thèse de E. ROYER [82].

L'algorithme est basé sur l'utilisation de tirages aléatoires. A partir d'un échantillon de 5 points en correspondance dans les images 0,1 et 2, on commence par calculer la matrice essentielle $\mathtt{E} = [\mathbf{t}]_\times \mathtt{R}$ entre les caméras 0 et 2 avec l'algorithme des 5 points. On en déduit le déplacement $(\mathtt{R}, \mathbf{t})$ de la caméra entre ces deux vues et on triangule les 5 points pour obtenir leur position 3D à l'aide de la méthode décrite dans la section 1.3.2. Ces points 3D sont ensuite utilisés pour calculer la pose de la caméra 1, avec les correspondances 3D/2D et la méthode de GRÜNERT expliquée dans la section 1.3.3. Chaque tirage aléatoire de 5 points nous a ainsi donné une hypothèse pour la pose des trois caméras. Pour chaque hypothèse, on reconstruit l'ensemble des

points appariés avec la même méthode de triangulation et on compte ceux qui sont compatibles avec la géométrie qui vient d'être calculée. On utilise pour cela un seuil (par exemple 2 pixels) sur l'erreur de reprojection dans les images. L'hypothèse pour laquelle on obtient le plus grand nombre de points cohérents est retenue (RANSAC).

Finalement, les 3 poses calculées et les coordonnées des points triangulés sont ré-estimées plus précisément par un ajustement de faisceaux standard.

3.4 Méthode de reconstruction 3D incrémentale

Le calcul du mouvement de la caméra et de la structure de la scène est réalisé de manière incrémentale et récursive. Une fois l'initialisation terminée, le nuage de points reconstruits en 3D permet de localiser la caméra pour l'image qui suit immédiatement le premier triplet d'images clef. L'opération est répétée, et on localise de la même manière la caméra pour les images suivantes, jusqu'à ce qu'on introduise une nouvelle image clef. Cette image clef sera utilisée pour reconstruire de nouveaux points 3D et ainsi compléter le nuage obtenu. Ensuite, on pourra de nouveau calculer la pose de la caméra pour les images suivantes avec les points récemment ajoutés, et ainsi de suite. Ce sont les points reconstruits à partir des images clef 0 à $i-1$ et visibles dans l'image courante qui permettent de calculer la pose de la caméra pour les images de la vidéo jusqu'à l'image clef suivante i.

3.4.1 Calcul de la pose courante

Supposons que les poses des caméras C^0 à C^{i-1} correspondant aux images clef I^0 à I^{i-1} ont été calculées dans le repère de référence. On a également reconstruit un ensemble de points 3D à partir de leurs observations en correspondance dans les images. L'étape suivante est le calcul de la pose courante C de la caméra à l'image la plus récemment acquise. Pour cela, on met en correspondance l'image courante I avec l'image I^{i-1} (dernière image clef sélectionnée) afin de déterminer un ensemble de points $\mathcal{P}$ visibles dans l'image I et dont les coordonnées 3D ont déjà été estimées. On utilise le calcul de pose 3 points de Grünert et un processus par tirages de type RANSAC pour déterminer la pose C à partir de l'ensemble des correspondances 3D/2D. Cette pose est ensuite calculée plus précisément à l'aide d'une optimisation non-linéaire par Levenberg-Marquardt des 6 paramètres de pose.

Calcul de l'incertitude sur la pose

Une fois la pose calculée, il est intéressant de déterminer l'incertitude que l'on a dans le résultat. Celle-ci est utilisée dans l'algorithme de reconstruction : tant que la confiance dans la localisation de la caméra avec les points disponibles est bonne, on peut passer à l'image suivante, sinon il peut être nécessaire de reconstruire de nouveaux points. D'autre part, l'incertitude associée à la pose pourrait être utile si l'on souhaitait fusionner des informations de position provenant de capteurs différents pour améliorer la précision de la localisation.
La propagation complète des incertitudes depuis l'initialisation de la reconstruction 3D jusqu'à la localisation est très coûteuse en temps de calcul. Pour pouvoir calculer l'incertitude de localisation en temps-réel, on fait l'hypothèse que tous les points 3D utilisés dans le calcul de pose ont été reconstruits avec la même précision. Ensuite, la matrice de covariance Cov des paramètres de pose s'obtient directement via l'inverse de la matrice Hessienne approchée à la fin de la minimisation. On calcule un ellipsoïde de confiance à 90% pour le centre x de la caméra par $\Delta x^T Cov^{-1} \Delta x \leq 6.25$ (puisque x suit une loi gaussienne de covariance connue Cov, $\Delta x^T Cov^{-1} \Delta x$ obéit à une loi du $\mathcal{X}_2$ à 3 degrés de liberté). Une autre méthode plus judicieuse (avec des hypothèses moins fortes) et probablement possible en temps-réel (basée sur [84]) aurait été de considérer les incertitudes sur les points reconstruits mais de négliger les covariances couplées ainsi que les incertitudes des poses de caméra précédentes.

3.4.2 Sélection des images clef et triangulation de points

Comme on l'a vu précédemment, et pour maximiser les angles de triangulation, toutes les images de la séquence vidéo ne sont pas utilisées pour la reconstruction de la structure 3D de la scène mais uniquement un "sous-échantillon" d'images clef. Pour chaque image, la voie normale est d'estimer la pose de la caméra dans le repère de reconstruction. Ensuite on calcule un critère qui indique si, oui ou non, une image doit être ajoutée en tant que nouvelle image clef. Dans l'application développée, le critère prend en compte plusieurs paramètres :

- D'abord, si le nombre de points en correspondance avec la dernière image clef I^{i-1} est inférieur à un seuil M fixé au préalable ($M = 200$ à 400 dans nos expérimentations), alors une nouvelle image clef est demandée.

- Il est également nécessaire que l'incertitude sur la pose calculée ne soit pas trop élevée (par exemple que le grand axe de l'ellipsoïde de confiance ne dépasse pas la distance moyenne entre deux images clef successives).
- On vérifie également que le nombre de points en correspondance 2D/3D satisfaisant la géométrie de la pose (*inliers*) est suffisant (supérieur à un nombre limite $n_l = 10$).

Si toutes les conditions sont respectées, l'ajout d'une nouvelle image clef n'est pas demandé et on passe à l'image suivante pour calculer de nouveau la pose de la caméra et le critère. Au contraire, si l'une de ces conditions n'est pas satisfaite, alors on ajoute une nouvelle image clef. Bien sûr, afin de disposer de suffisamment d'appariements fiables, l'image qui a refusé le critère de sélection n'est pas choisie comme la nouvelle image clef I^i mais c'est l'image qui précède immédiatement qui est choisie. De nouveaux points de la scène (c'est-à-dire ceux qui ne sont observés jusqu'à maintenant que dans les images I^{i-2}, I^{i-1} et I^i) sont reconstruits en trois dimensions par une triangulation à partir de trois vues (section 1.3.2).

La Figure 3.4 montre l'évolution de ces trois critères sur une séquence de 200 images. Les lignes verticales indiquent qu'une image clef a été choisie, et les petits cercles montrent quel critère a déclenché l'ajout d'une image clef. On peut remarquer que c'est le premier critère, sur le nombre total de points en correspondance avec la dernière image clef, qui est utilisé le plus souvent. Sur la plupart des séquences traitées, ce critère seul s'avère suffisant pour la réussite de l'algorithme. Cependant, l'ajout de deux critères supplémentaires permet d'apporter de la robustesse dans les cas critiques où le nombre de faux appariements de points est important.

3.5 Reconstruction 3D par ajustement de faisceaux local

3.5.1 Principe et mise en oeuvre

Lorsque la i-ème image clef I^i est sélectionnée et la caméra C^i ajoutée à la reconstruction, une étape d'optimisation par ajustement de faisceaux est effectuée. On utilise l'algorithme de Levenberg-Marquardt pour minimiser la fonction de coût $f^i(\mathcal{C}^i, \mathcal{P}^i)$ où $\mathcal{C}^i$ et $\mathcal{P}^i$ sont respectivement les paramètres extrinsèques des caméras et les coordonnées 3D des points choisis pour cette étape i. L'idée importante de l'approche présentée est de réduire l'ensemble des paramètres optimisés aux plus récemment calculés. Pour cela,

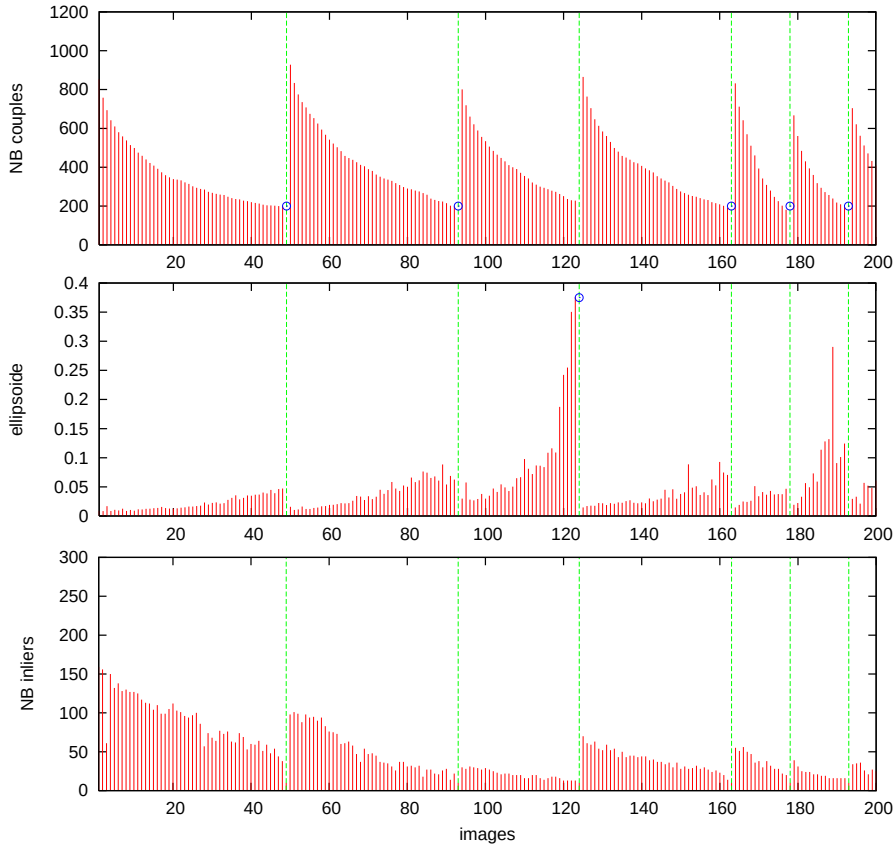


FIG. 3.4 – Évolution des critères de sélection des images clef : nombre de couples de points appariés, ellipsoïde d'incertitude sur la pose calculée et nombre d'*inliers* sur une séquence de 200 images.

on optimise les paramètres extrinsèques des n dernières caméras et les points qui s'y projettent sur une sous-séquence de N images (avec $N > n$). Cela signifie qu'on affine les coordonnées des points 3D et les poses des n caméras en minimisant une fonction de coût qui tient compte des reprojections 2D sur N images (voir Figure 3.5).
Ainsi, on a $\mathcal{C}^i = \{C^{i-n+1} \dots C^i\}$ et $\mathcal{P}^i$ contient tous les points 3D se projetant dans les caméras de $\mathcal{C}^i$. La fonction de coût f^i est la somme des erreurs de reprojections des points de $\mathcal{P}^i$ dans les images I^{i-N+1} à I^i :

$$f^i(\mathcal{C}^i, \mathcal{P}^i) = \sum_{C^k \in \{C^{i-N+1} \ \dots \ C^i\}} \sum_{\mathbf{P}_j \in \mathcal{P}^i} \left(\epsilon_j^k\right)^2 \tag{3.1}$$

où $\left(\epsilon_j^k\right)^2 = \left\|\mathbf{p}_j^k - \pi(\mathtt{C}^k\bar{\mathbf{P}}_j)\right\|^2$ est la distance euclidienne au carré entre $\mathtt{C}^k\bar{\mathbf{P}}_j$, projection estimée du point $\mathbf{P}_j$ par la caméra C^k et le point détecté $\mathbf{p}_j^k$ correspondant.
Comme on vient de le voir, les paramètres n (nombre de caméra optimisées à chaque étape) et N (nombre d'images prises en compte dans la fonction de reprojection) sont les deux principaux paramètres impliquées dans le processus d'optimisation. Le choix de leur valeur peut influencer la précision des résultats et le temps de calcul. Les expériences ont permis de déterminer les valeurs de n et N (typiquement, on prend $n = 3$ et $N = 10$) qui fournissent des résultats précis.
Dans notre implémentation, chaque optimisation par ajustement de faisceaux local comporte en fait deux étapes d'optimisation par la méthode de Levenberg-Marquardt (Algorithme 1 de la section 1.5), séparées par une mise à jour des points 3D et 2D satisfaisant la géométrie (le j-ème point 3D et la i-ème pose clef sont consistants si l'erreur ϵ_j^i est inférieure à un seuil $\epsilon = 2$ *pixels*). Une série est arrêtée si l'erreur ne descend pas suffisamment entre deux itérations ($||\boldsymbol{\epsilon}_k|| \geq \alpha||\boldsymbol{\epsilon}_{k-1}||$), avec $\alpha = 0.9999$) ou si le nombre maximum d'itérations est atteint (5 dans notre cas). En pratique, le nombre d'itérations nécessaire pour chaque ajustement de faisceaux local à l'étape i est assez faible. Cela est dû au fait que, mise à part la dernière caméra ajoutée, toutes les poses de caméras ont déjà été optimisées aux étapes $i-1$, $i-2$, ...

On peut noter qu'au début de la reconstruction, on ne raffine pas seulement les paramètres de la fin de la séquence mais la structure 3D toute entière. Ainsi, pour $i \leq N_f$, on prend $N = n = i$. N_f est le nombre maximum de caméras de la séquence pour lequel l'optimisation à l'étape i soit globale. Passée cette limite, l'ajustement de faisceau est local (dans nos expériences, on choisit $N_f = 20$). De cette manière, les données initiales sont optimales, ce qui est important étant donné la nature récursive de l'algorithme. De plus, cela ne pose pas de problème de temps de calcul puisque le nombre de paramètres est encore relativement restreint.

3.5.2 Complexité de calcul pour l'ajustement de faisceaux local et global

La complexité de calcul d'une itération d'ajustement de faisceaux a été donnée en 1.5.6. Rappelons qu'elle s'écrit

$$\Theta(N_r + p.N_c^2 + N_c^3).$$

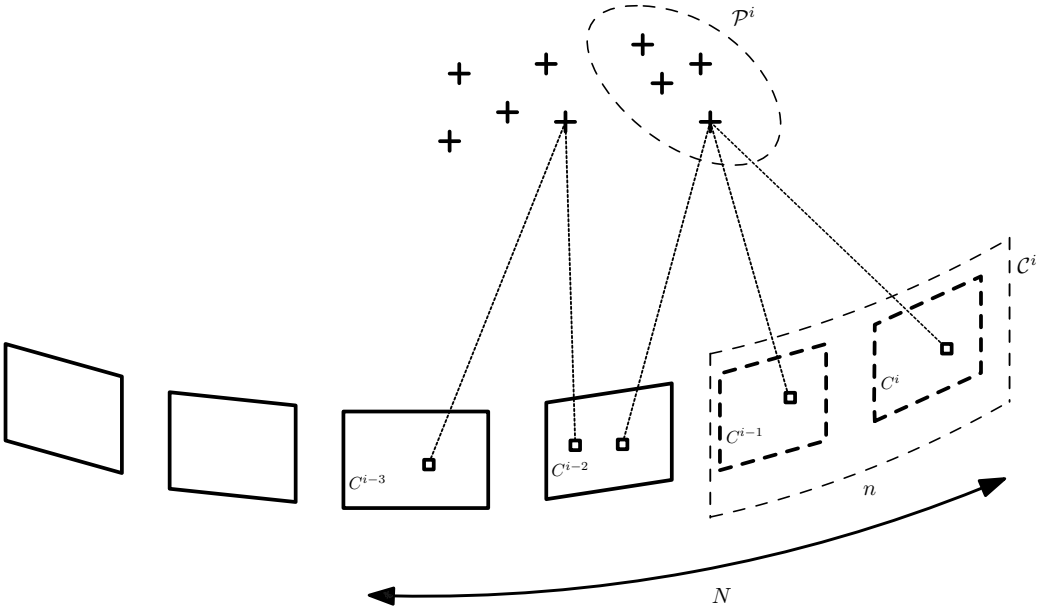


FIG. 3.5 – Ajustement de faisceaux local appliqué lorsque la i-ème image clef (de pose C^i) est sélectionnée. Seuls les points $\mathcal{P}^i$ et poses $\mathcal{C}^i$ entourées sont optimisées. La fonction de coût minimisée tient compte des projections des points 3D dans les N dernières images clef. Ici, $n = 2$ et $N = 4$.

avec p nombre moyen de points se projetant dans une image, N_c nombre de poses de caméras optimisées, et N_r nombre de reprojections 2D prises en compte dans la minimisation.
Pour une séquence vidéo de N_{seq} images clef, la complexité d'une itération d'ajustement de faisceaux appliqué aux n dernières caméras et prenant en compte les reprojections 2D dans N images est

$$\Theta(p.N + p.n^2 + n^3) \tag{3.2}$$

alors que la complexité d'une itération d'ajustement de faisceaux global sur la séquence complète est

$$\Theta\left(p.N_{seq}^2 + N_{seq}^3\right) \tag{3.3}$$

puisque le nombre N_r de reprojections 2D prises en compte, qui est de l'ordre du nombre total ($p.N_{seq}$) de reprojections 2D de la séquence, est négligeable devant $p.N_{seq}^2$.
Il est clairement avantageux de diminuer la valeur des paramètres (n et N) si l'on souhaite réduire la complexité de calcul de l'optimisation. Par exemple, le gain en complexité obtenu par rapport à l'ajustement de faisceaux global

pour une séquence de 20 images clef et 150 reprojections 2D par image est donné dans le Tableau 3.1. Le gain est de 10 pour $n = 5, N = 20$ et de 25 pour $n = 3, N = 10$. La diminution en complexité de calcul est également illustrée par la Figure 3.6 où l'on voit la réduction de la dimension des matrices Jacobienne $\mathtt{J}$ et Hessienne approchée $\mathtt{J}^\top\mathtt{J}$ impliquée dans la résolution du système linéaire 1.51 pour une itération d'ajustement de faisceaux.

Type	p	n	N	gain
global	150	20	20	**1**
local 1	150	5	20	**10**
local 2	150	3	10	**25**

TAB. 3.1 – Gain en complexité de calcul obtenu avec l'ajustement de faisceaux local par rapport à l'ajustement de faisceaux global pour une itération.

Cette étude porte sur une itération de l'algorithme de LEVENBERG-MARQUARDT. En pratique, pour la méthode incrémentale, les paramètres n and N sont fixés et le nombre d'itérations est limité pour chaque ajustement de faisceaux local (on remarque que le nombre d'itérations nécessaire est assez faible). La complexité finale de notre méthode de reconstruction pour une séquence complète est linéaire avec N_{seq} alors qu'une seule itération d'ajustement de faisceau global croît avec N_{seq}^3. La méthode incrémentale est ainsi très intéressante en terme de complexité de calcul par rapport à une méthode classique avec ajustement de faisceaux global. Elle est d'autant plus efficace que la séquence est longue avec un nombre d'images élevé.

3.6 Résumé de l'algorithme

En résumé, l'approche proposée se déroule de la manière suivante :

1. Choisir un premier triplet d'images qui deviennent les 3 premières images clef (section 3.3). Fixer le repère global, estimer le mouvement relatif de la caméra, et reconstruire les points 3D.
2. Pour chaque image de la vidéo, calculer les mises en correspondance de points avec la dernière image clef (section 3.2) et estimer la pose de la caméra et son incertitude (section 3.4.1). Déterminer si une nouvelle image clef doit être sélectionnée (section 3.4.2). Si non, répéter à partir de 2.

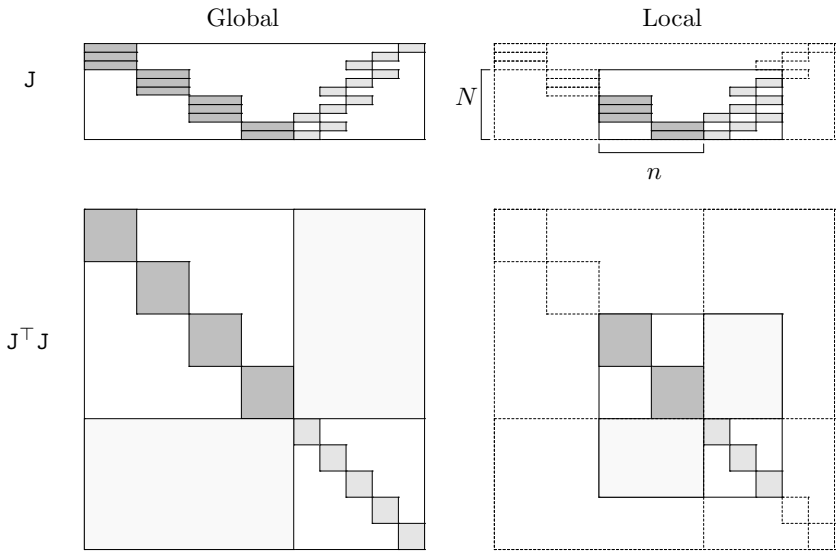


FIG. 3.6 – Ajustement de faisceaux local : réduction de la dimension de la matrice Jacobienne J et de la matrice Hessienne approchée $\mathtt{J}^\top\mathtt{J}$ (4 caméras, 5 points, $n = 2$ et $N = 3$).

3. Si une nouvelle image clef doit être ajoutée, sélectionner l'image qui précède immédiatement comme la nouvelle image clef, trianguler des nouveaux points 3D et procéder à l'ajustement de faisceaux local (section 3.5). Répéter à partir de 2.

3.7 Résultats et performances

Afin de valider la méthode présentée dans ce chapitre, nous avons réalisé plusieurs séries d'expériences sur des données réelles. Une première série a été menée sur le campus des Cézeaux, près du LASMEA à Clermont-Ferrand, avec un véhicule expérimental *Cycab* (Figure 3.11(a)). Le *Cycab* est un véhicule électrique "intelligent" équipé de plusieurs capteurs de localisation comme un récepteur GPS ou encore des capteurs proprioceptifs pour l'odométrie des roues. Le capteur qui nous intéresse le plus est la caméra embarquée située sur le toit du véhicule et dirigée vers l'avant, fournissant des images de taille 1027×768 pixels qui sont sous-échantillonnées et traitées à la résolution de 512×384. Notre but est bien sûr d'analyser ces images pour en extraire des informations géométriques sur la trajectoire du véhicule et sur l'environnement. Ces expériences ont permis de tester l'approche, de mesurer

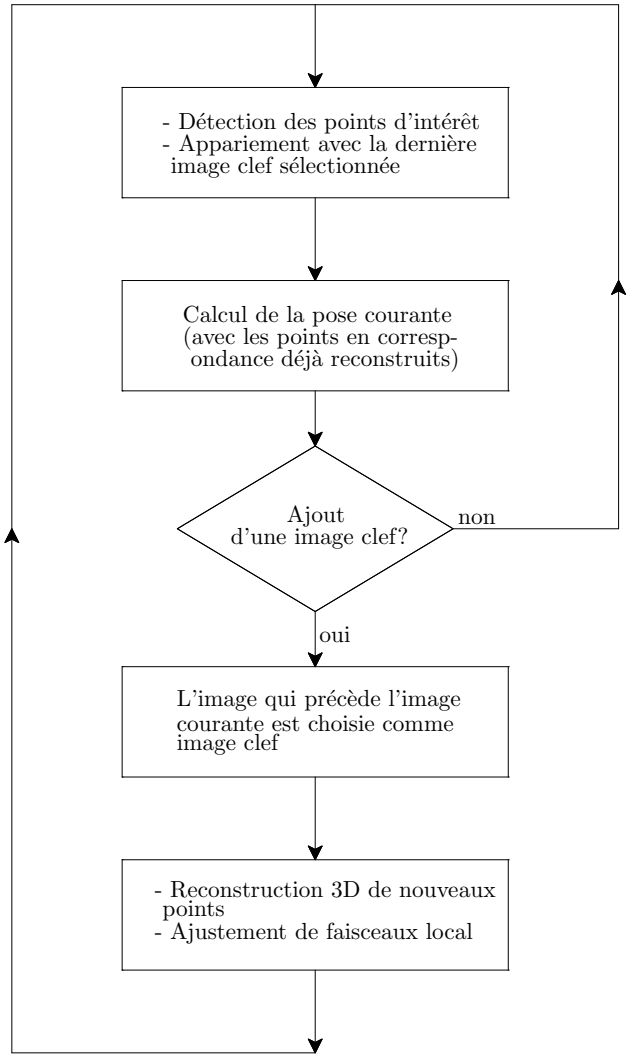
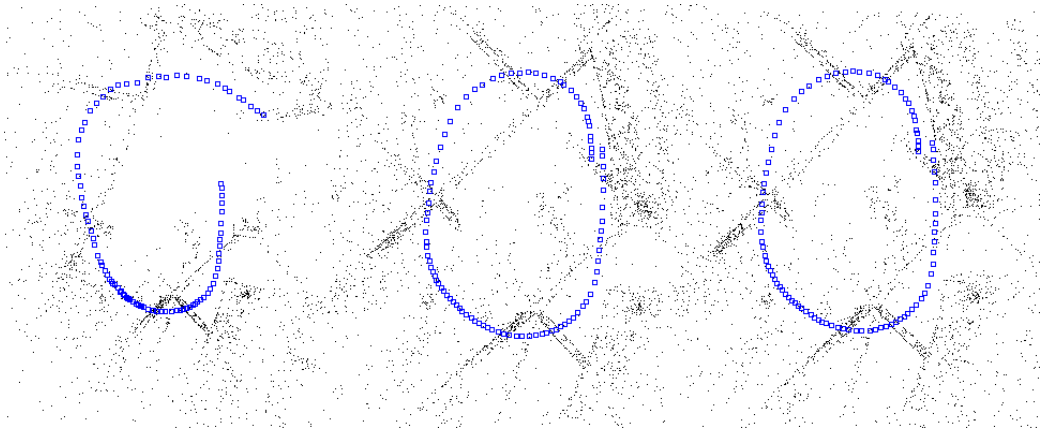


FIG. 3.7 – Diagramme résumant le fonctionnement de la méthode après l'initialisation.

ses performances et de la comparer à une autre méthode de vision, ainsi qu'à la vérité terrain fournie par le récepteur GPS différentiel. Une deuxième série d'expériences a été effectuée avec le même type de caméra, mais embarquée sur une automobile en milieu urbain dense dans les centres-ville de Clermont-Ferrand et Compiègne. Finalement, des expériences complémentaires ont été réalisées afin de tester plusieurs points techniques de l'algorithme.

3.7.1 Comparaison avec l'ajustement de faisceaux global (méthode de reconstruction hiérarchique)

La Figure 3.8(a) montre les résultats obtenus à partir d'une séquence de 93 images clef acquise par la caméra embarquée sur le *Cycab* (Séquence "boucle"). Des images de la séquence sont visibles sur la Figure 3.8(b). La trajectoire réelle est fermée et se boucle sur elle-même. On peut alors voir la "dérive" de la reconstruction lorsque les paramètres n et N sont mal choisis. La dernière reconstruction est obtenue avec une méthode de reconstruction



(a) Les 2 premières images représentent les reconstructions incrémentales de la séquence "boucle" avec $n = 3, N = 3$ ($RMS = 0.839$ pixels) puis $n = 3, N = 10$ ($RMS = 0.616$ pixels). La dernière représente une reconstruction hiérarchique (RMS=0.589 pixels).

(b) 3 images tirées de la séquence.

FIG. 3.8 – Séquence "boucle". (a) exemples de reconstruction (vue de dessus). (b) images de la vidéo.

considérée comme très précise. Elle est basée sur un algorithme qui fusionne

des sous-séquences avec un ajustement de faisceaux appliqué de manière hiérarchique [83]. La dernière étape de cet algorithme consiste en une optimisation globale de tous les paramètres 3D (ajustement de faisceaux global). Cette méthode est prise comme référence pour l'évaluation des nos résultats.

Par la suite, nous comparons une reconstruction générée par notre méthode incrémentale avec une reconstruction obtenue par la méthode de référence pour une séquence de 112 images clef. Le sous-échantillonnage réalisé est le même avec les deux méthodes afin de n'évaluer que l'aspect reconstruction 3D de l'algorithme. Le nombre de points 3D reconstruits est d'environ 14.900 et la longueur de la trajectoire est de 80 mètres (Figure 3.9(a)). Au cours de nos expériences, nous avons fait varier les paramètres n et N qui correspondent respectivement au nombre de caméras optimisées et au nombre de caméras prises en compte à chaque étape, lors de l'ajustement de faisceaux local. Le but est d'étudier l'influence des valeurs données à ces paramètres sur la qualité des résultats. Pour cela, nous évaluons l'erreur de reprojection et l'erreur de positionnement des caméras par rapport à la référence.

Comparaison de l'erreur de reprojection

L'erreur globale de reprojection dans les images des points 3D estimés est mesurée par le coefficient RMS (section 1.4.2). Les résultats exprimés en pixels sont recensés dans le Tableau 3.10(a) de la Figure 3.10. Les valeurs vont de 0.978 pixels pour la plus "mauvaise" reconstruction à 0.602 pixels pour la "meilleure". Notons que le RMS mesuré pour une reconstruction hiérarchique est de 0.589 pixels. Il faut également noter qu'il n'est pas du tout intéressant de choisir N proche de n, même si n est grand. Par exemple, si on prend $N = n$, le Tableau 3.10(a) (première colonne) montre que plus n est grand, plus l'erreur résultante est importante. Cela provient du fait que l'on ne tient pas assez compte de l'antériorité de la structure (alors qu'elle est sûre) lors de l'optimisation. On peut l'expliquer de la façon suivante. Deux parties de la séquence de N caméras sont impliquées dans la fonction de coût : une première partie contenant $N - n$ caméras est fixée, une seconde contenant n caméras est optimisée. Les deux parties sont liées par les points 3D eux-mêmes optimisés. La seconde partie est contrainte par les points en commun, eux-mêmes contraints par la première partie. Puisque chaque point est fixé par au moins deux caméras de la première partie, la contrainte $N \geq n + 2$ (plus forte) est nécessaire.
Si on choisit $N = n$ ou $N = n + 1$, alors l'ajustement de faisceau local donne une solution dans un repère qui n'est pas fixé (position et échelle) par rapport

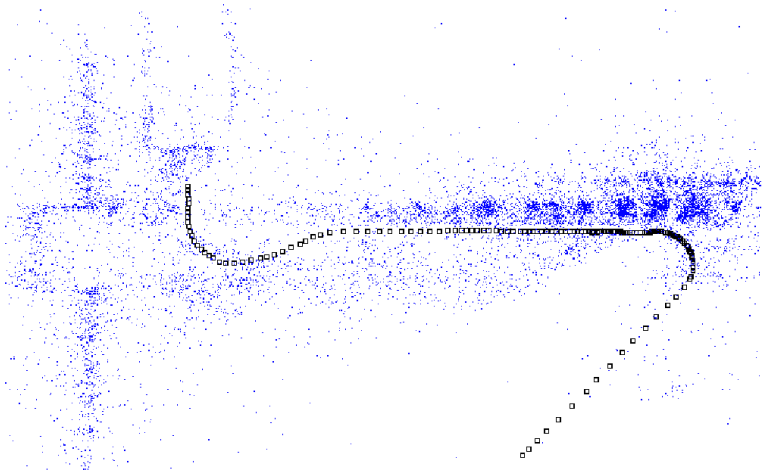
au début de la reconstruction. Même si cette solution est optimale localement, elle n'est pas assurée d'être bonne globalement. On peut donc obtenir une forte dérive de la trajectoire comme on peut le voir sur la Figure 3.8(a) et une modification du facteur d'échelle au cours de la reconstruction. L'approche d'ajustement de faisceaux local présentée par ZHANG et SHAN [108] basée sur l'optimisation d'un triplet de caméras ne résout pas ce problème.

Erreur de positionnement des caméras

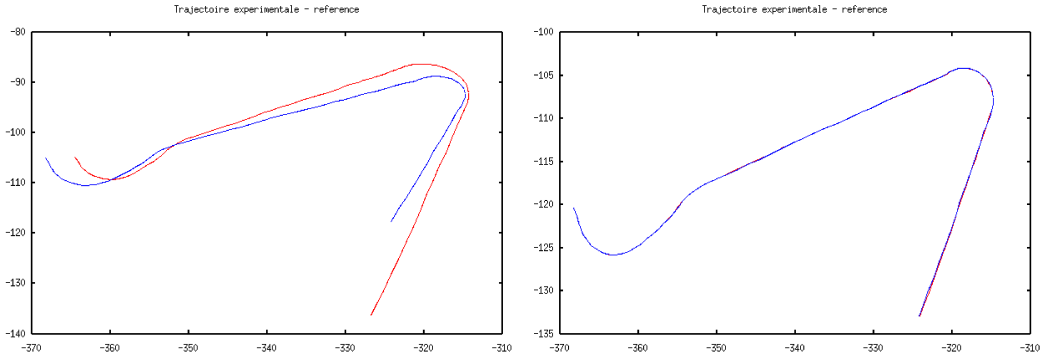
Dans un but de comparaison, la trajectoire calculée a été recalée à la séquence de référence reconstruite par la méthode hiérarchique. Pour cela on lui a appliqué une transformation rigide (rotation, translation, et mise à l'échelle) pour l'exprimer dans le même repère. L'approche est décrite par FAUGERAS et HEBERT [20]. Le repère de référence étant gradué en mètres, on peut mesurer l'erreur moyenne de position des caméras (en mètres) donnée par $\frac{\sum_i \sqrt{{E_x^i}^2 + {E_y^i}^2 + {E_z^i}^2}}{N_{seq}}$ où E_x^i, E_y^i, E_z^i sont les erreurs en x, y et z de la position de la caméra i par rapport à la même caméra dans la reconstruction de référence. Les résultats en fonction de n et N sont donnés dans le Tableau 3.10(b) et la Figure 3.10(d). On peut remarquer que l'erreur moyenne est inférieure à 13 cm si $N \geq n + 3$ $\forall n$, et même inférieure à 11 cm à 4 exceptions près. Cela représente une erreur relative très faible étant donné la longueur des déplacements (0.13 à 0.16% de la distance parcourue). La Figure 3.9(b) quant à elle, montre d'abord une reconstruction trop déformée où le recalage est médiocre et ensuite une bonne reconstruction. Comme pour l'erreur de reprojection, les résultats sont mauvais avec $N = n$ ou $N = n + 1$ et satisfaisants avec $N \geq n + 2$.

3.7.2 Comparaison avec la vérité terrain (GPS différentiel)

Les récepteurs GPS sont très répandus de nos jours dans les systèmes d'aide à la navigation. Les produits grand public fournissent une position avec une précision de l'ordre de 3 à 4 mètres. Le système différentiel quant à lui utilise une base au sol (en plus des satellites) dont la position est connue précisément et envoie des corrections aux récepteurs. Avec cette technique, on obtient une position beaucoup plus précise. Par exemple, le matériel de localisation GPS installé dans le *Cycab* (Real Time Kinematics Differential GPS, modèle Thalès Sagitta) fournit une position en temps-réel avec une



(a) Vue de dessus de la reconstruction de référence.



(b) 2 exemples de recalage (gauche : $n = 3, N = 3$. droite $n = 3, N = 10$).

FIG. 3.9 – Comparaison avec la reconstruction hiérarchique de référence.

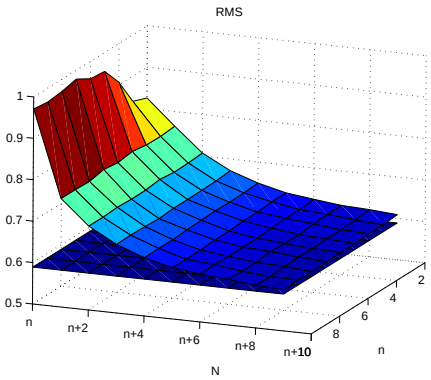
précision de l'ordre de 2 cm dans le plan horizontal et 20 cm sur l'axe vertical. Le but de ces expériences est d'observer le comportement de la méthode de reconstruction dans un environnement réel complexe et de comparer la trajectoire obtenue à celle donnée par le capteur GPS, prise comme vérité terrain. La caméra est installée sur le *Cycab* (Figure 3.11(a)) dont la vitesse est ici d'environ 1.1 m/s. La distance parcourue est de 70 m et la séquence vidéo d'une durée de 1 min (taille d'image 512×384 $pixels$ à 7.5 $images/s$). Plus de 4.000 points 3D ont été reconstruits et 94 images sélectionnées comme images clef à partir d'une série de 445 images au total. Cette séquence est particulièrement intéressante de par son contenu (piétons passant devant la caméra, soleil, etc...) qui n'est pas favorable au processus de reconstruction. De plus, l'environnement (semi-urbain) est beaucoup plus approprié à une

n \ N	n	$n+1$	$n+2$	$n+3$	$n+4$	$n+5$	$n+6$	$n+7$	$n+8$
$n=2$	0.825	0.764	0.707	0.671	0.649	0.632	0.624	0.616	0.613
$n=3$	0.839	0.761	0.704	0.668	0.646	0.632	0.620	0.616	0.610
$n=4$	0.906	0.762	0.703	0.667	0.645	0.629	0.621	0.615	0.608
$n=5$	0.954	0.768	0.707	0.666	0.646	0.629	0.619	0.611	0.608
$n=6$	0.957	0.762	0.703	0.666	0.645	0.626	0.616	0.610	0.605
$n=7$	0.978	0.767	0.703	0.663	0.643	0.626	0.615	0.613	0.607
$n=8$	0.970	0.760	0.704	0.665	0.641	0.623	0.615	0.608	0.605
$n=9$	0.966	0.761	0.702	0.666	0.641	0.622	0.617	0.608	0.602
$n=10$	0.970	0.761	0.702	0.665	0.640	0.624	0.616	0.610	0.603

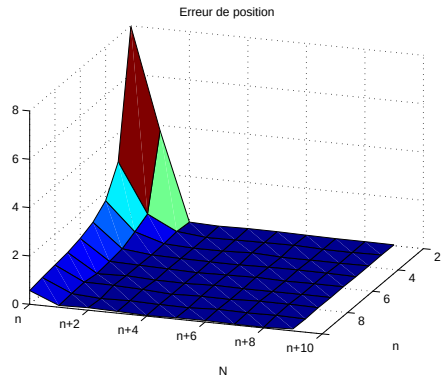
(a) RMS (en pixels) en fonction de n et N.

n \ N	n	$n+1$	$n+2$	$n+3$	$n+4$	$n+5$	$n+6$	$n+7$	$n+8$
$n=2$	7.966	4.581	0.112	0.085	0.114	0.105	0.075	0.050	0.045
$n=3$	2.783	0.957	0.200	0.054	0.073	0.093	0.104	0.050	0.035
$n=4$	1.600	0.637	0.124	0.051	0.055	0.109	0.039	0.044	0.029
$n=5$	1.157	0.271	0.099	0.064	0.046	0.041	0.069	0.081	0.071
$n=6$	0.942	0.170	0.112	0.082	0.032	0.074	0.102	0.055	0.073
$n=7$	0.803	0.088	0.131	0.055	0.036	0.088	0.066	0.071	0.070
$n=8$	0.754	0.189	0.177	0.127	0.027	0.076	0.063	0.117	0.114
$n=9$	0.659	0.148	0.057	0.097	0.051	0.026	0.097	0.065	0.093
$n=10$	0.563	0.085	0.124	0.093	0.028	0.058	0.068	0.037	0.027

(b) Erreur (en mètres) en fonction de n et N.



(c) Visualisation 3D du Tableau (a). Le plan horizontal représente le niveau de RMS résultant de la reconstruction hiérarchique de référence.



(d) Visualisation 3D du Tableau (b).

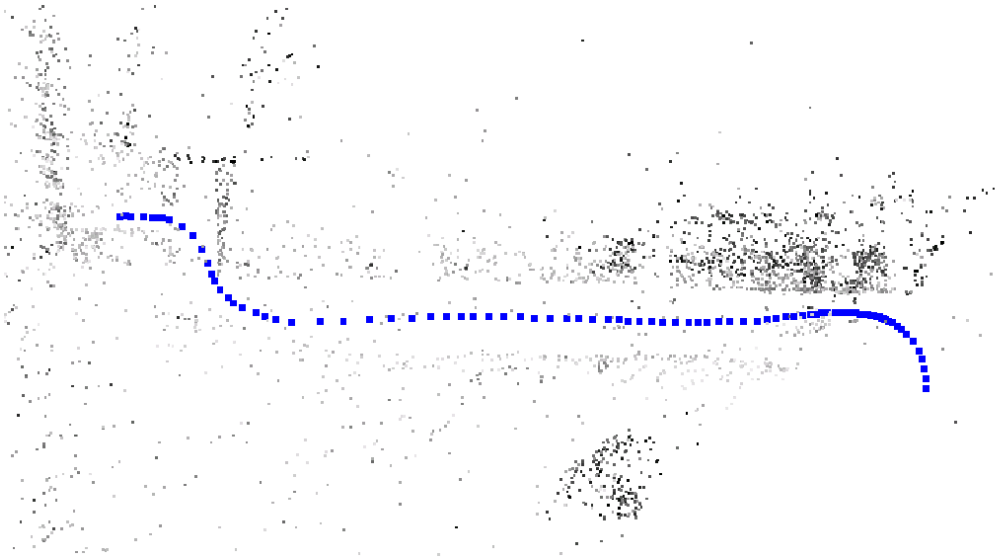
FIG. 3.10 – Erreur de reprojection et erreur moyenne sur la position des caméras par rapport à la séquence de référence en fonction des paramètres n et N.

localisation par GPS puisque les satellites ne sont pas occultés par de hauts immeubles. Des images de cette séquence sont visibles sur la Figure 3.11(b).

Pour comparer la trajectoire obtenue avec notre algorithme à celle fournie

(a) Le *Cycab*

(b) 2 images tirées de la séquence "GPS". On note la présence de piétons.



(c) Vue de dessus de la reconstruction 3D ($\approx$ 4.000 points 3D et 94 positions clef).

FIG. 3.11 – Séquence "GPS" : (a) le *Cycab*, (b) images de la vidéo, (c) reconstruction vue de dessus.

par le capteur GPS, la transformation rigide évoquée précédemment dans la section 3.7.1 a été appliquée pour que les deux trajectoires soient exprimées dans le même repère métrique. Il est alors facile de mesurer les erreurs de positions 3D ou les erreurs 2D (en *mètres*) dans le plan horizontal. La Figure 3.12 montre le recalage de la trajectoire vision (pour $n = 3$ et $N = 10$) avec la référence GPS. L'erreur maximale mesurée est de 2.0 m avec une erreur 3D moyenne de 41 cm et une erreur 2D moyenne de moins de 35 cm.

Comme mentionné auparavant, l'optimisation est appliquée aux n dernières poses de caméra estimées, en tenant compte des reprojections sur un plus grand nombre N d'images. Nous avons ainsi testé plusieurs valeurs de n et N, comme indiqué dans le Tableau 3.2. Il faut rappeler que N doit être

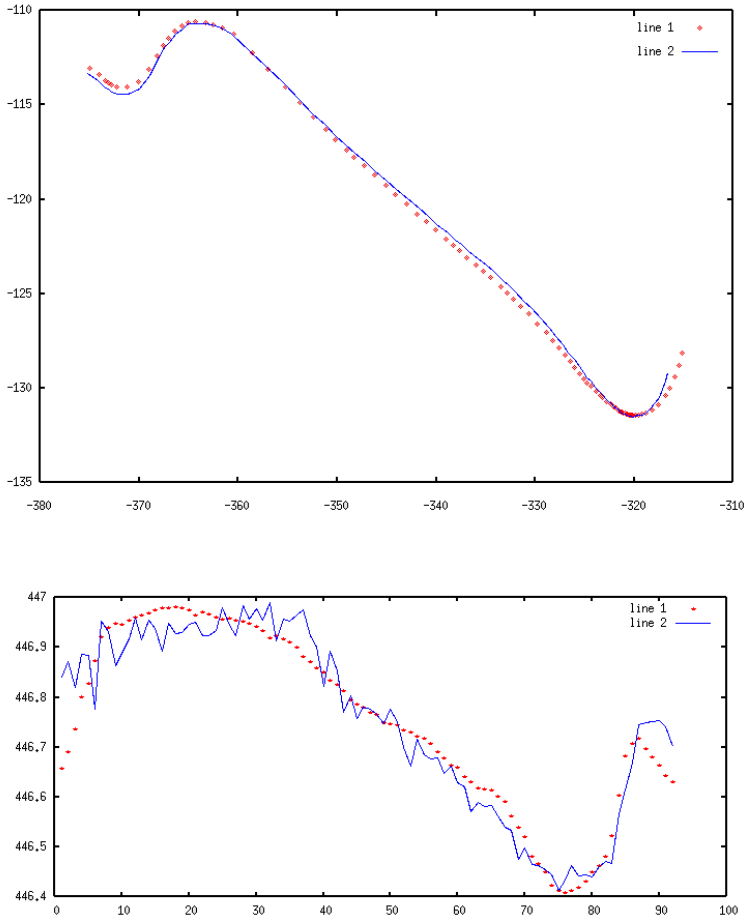


FIG. 3.12 – Recalage avec la trajectoire GPS de référence, haut : dans le plan horizontal, bas : suivant l'axe vertical. La ligne continue représente la trajectoire GPS et les points les positions estimées aux images clef. Les coordonnées sont exprimées en *mètres*.

supérieur ou égal à $n + 2$ pour fixer le repère et le facteur d'échelle de la partie optimisée. Pour $N = n$ ou $n + 1$, il est arrivé que le processus de reconstruction ait échoué avant la fin de la séquence. Cela confirme que la condition $N \geq n+2$ est importante. Si $n \geq 3$ et N d'autant plus grand, alors les résultats sont encore plus précis ; mais le temps de calcul peut devenir un problème, comme on le verra par la suite. Dans nos expériences, des résultats similaires et acceptables ont été obtenus avec $n = 3$ ou 4 et $6 \leq N \leq 10$ (valeurs en gras dans le Tableau 3.2).

n \ N	n	$n+1$	$n+2$	$n+3$	$n+5$	$n+7$
$n=2$	échec	échec	0.55	0.49	0.85	1.99
$n=3$	échec	3.28	0.45	**0.41**	**0.41**	**0.41**
$n=4$	6.53	1.77	0.42	**0.40**	**0.41**	**0.27**
global	0.33					

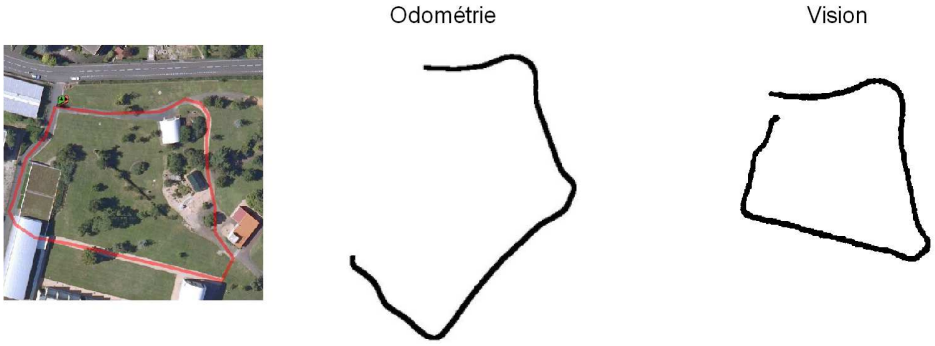
TAB. 3.2 – Erreur de position 3D moyenne (en *mètres*) de la méthode incrémentale pour différentes valeurs de n et N et de l'ajustement de faisceaux global par rapport à la référence GPS.

Autres exemples de trajectoires

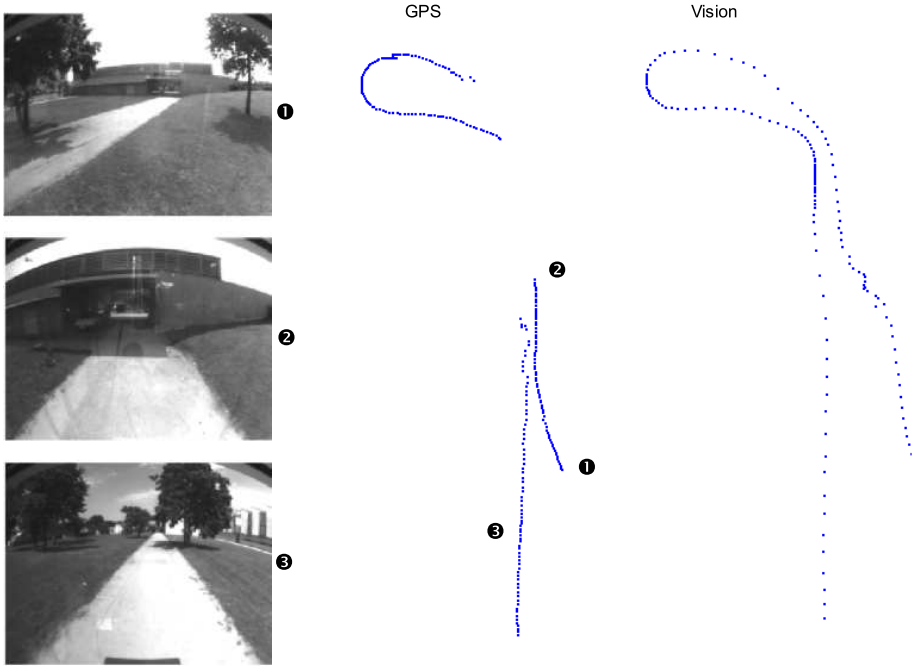
A titre d'exemple, la Figure 3.13 montre deux autres trajectoires réalisées avec le *Cycab*.
La première est une boucle complète dont la longueur est 342 m. La Figure 3.13(a) présente les résultats obtenus avec deux outils de localisation différents : vision monoculaire et odométrie des roues. On remarque une dérive dans les deux cas puisque le point d'arrivée ne correspond pas au point de départ, mais celle-ci est beaucoup moins importante dans le cas des calculs par vision. On voit bien ici l'intérêt d'utiliser le capteur caméra qui offre une précision supérieure à l'odométrie.
La seconde trajectoire, illustrée sur la Figure 3.13(b), mesure 184 m de long et comprend un demi-tour et deux passages sous un tunnel. Les positions données par le GPS et celles calculées par vision sont présentées. On remarque la perte de localisation par GPS lors des deux passages sous le tunnel, puisque les signaux provenant des satellites ne sont plus captés. En ce qui concerne la trajectoire par vision, on remarque que malgré une légère dérive (les deux passages sous le tunnel se font normalement au même endroit) la trajectoire est complète. Cette expérience illustre bien le fait que les méthodes sont complémentaires. Le GPS donne une bonne position globale dans les zones dégagées, alors que la méthode de vision calcule toujours une bonne trajectoire relative localement mais susceptible de dériver.

3.7.3 Temps de calcul

Les calculs ont été réalisés sur un ordinateur PC standard (pentium 4 à 2.8 GHz, 1Go de mémoire RAM fonctionnant à 800 MHz, système d'exploi-



(a) Exemple de trajectoire bouclée (342 m). Gauche : vue aérienne de la trajectoire, milieu : odométrie des roues, et droite : vision monoculaire.



(b) Exemple de trajectoire (184 m) avec deux passages sous un tunnel (GPS et Vision).

FIG. 3.13 – Exemples de trajectoires et comparaison avec les autres capteurs de localisation.

tation Linux). Comme on peut s'y attendre, les temps de calcul augmentent avec n. C'est ce qu'on peut voir dans le Tableau 3.3. En pratique, ceux-ci n'augmentent pas beaucoup avec N. Pour la séquence GPS, le temps de calcul nécessaire à la méthode de référence était supérieur à 15 min alors qu'il

a fallu à notre algorithme entre 55 s et 1 min 30, soit 11 à 16 fois moins de temps. Ce gain en temps de calcul sera d'autant plus important que la trajectoire sera longue.

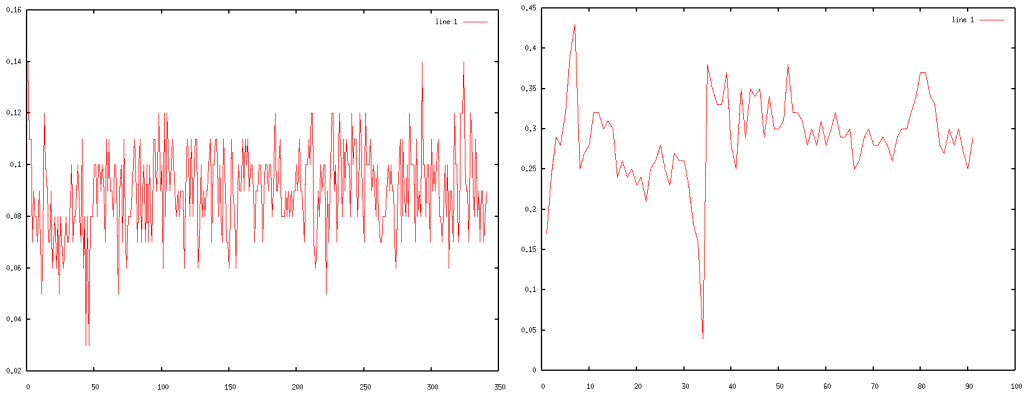
n	2	3	4	5	6
Temps d'optimisation (s)	0.22	0.28	0.30	0.35	0.41
Facteur de gain	15.8	13.9	13.4	12.0	11.2

TAB. 3.3 – Temps de calcul en fonction de n. (a) temps moyen passé dans l'ajustement de faisceaux, (b) facteur de gain par rapport à la méthode de référence.

Les expériences nous ont permis de déterminer que les valeurs $n = 3$ et $N = 10$, c'est-à-dire l'optimisation d'un triplet de caméras sur une sous-séquence de 10 images, donnent un bon compromis entre précision (Figure 3.10 et Tableau 3.2) et vitesse de calcul (Tableau 3.3). Dans les prochains exemples de reconstruction, ce sont ces valeurs qui seront choisies. Par exemple, les temps de calcul au long de la séquence avec ces paramètres sont donnés sur la Figure 3.14 et résumés dans le Tableau 3.4(a). Les temps mesurés incluent la détection des points d'intérêt ($\approx$ 1500 points de HARRIS par image), mise en correspondance, et calcul de pose pour chaque image. Les temps de calcul détaillés pour une image normale de chacune de ces opérations sont donnés dans le Tableau 3.4(b) et on remarque que c'est la phase d'appariement des points qui demande le plus de temps (55 ms). En ce qui concerne les images clef, les temps de calcul sont nettement supérieurs car il faut ajouter la reconstruction des points 3D et l'ajustement de faisceaux local. Les temps moyens mesurés et reportés dans le Tableau 3.4(a) sont de 0.09 s pour les images normales et 0.28 s pour les images clef, en remarquant que le temps disponible entre deux images est de 0.133 s à 7.5 $images/s$. Nous obtenons ainsi une reconstruction à 7.7 $images/s$ de moyenne.

Pour résumer, on vient de voir que procéder de manière incrémentale sans optimiser l'ensemble des paramètres de la reconstruction à chaque étape ne dégrade que très faiblement la qualité (erreur de reprojection, positionnement des caméras) des résultats. On arrive ainsi à diminuer notablement les temps de calcul tout en conservant une très bonne précision. Cependant, même si on peut réduire le nombre de paramètres optimisés, il est important de minimiser un critère qui tient compte de l'impact des nouveaux paramètres sur la partie antérieure, déjà optimisée. Dans le cas contraire, on obtient une

solution qui est seulement localement optimale, ce qui peut entraîner une forte dérive de la trajectoire.



(a) (b)

FIG. 3.14 – Temps de calcul (en *secondes*) pour (a) les images normales et (b) les images clef.

(a) Temps pour les images normales et les images clef

Images	Temps Max (s)	Temps Moyen (s)	Total (s)
Normales	0.14	0.09	30.69
Images clef	0.43	0.28	26.29

(b) Temps détaillés pour une image normale

Opération	Temps (ms)
détection points	16
appariements	55
calcul de pose RANSAC	15
optimisation + covariance	<10

TAB. 3.4 – Résultats - Temps de calcul.

3.7.4 Longues séquences en milieu urbain

Nous présentons ici deux reconstructions réalisées à partir de vidéos acquises dans les centres-ville de Compiègne et Clermont-Ferrand. Les conditions sont plus proches des applications visées pour l'automobile en milieu

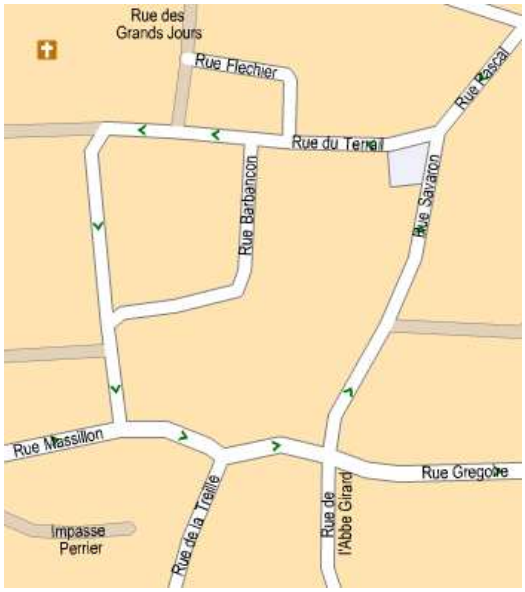
urbain dense. En effet, dans ce milieu, les signaux des satellites du système global GPS peuvent être masqués, ce qui entraîne un manque de précision du positionnement du véhicule : la localisation par vision peut alors s'avérer cruciale.
Au cours de ces expériences la vitesse approximative du véhicule se situe entre 8 et 13 km/h avec une fréquence d'acquisition de la caméra à 15 $images/s$. La distance totale parcourue est d'environ 700 m (Compiègne) et 400 m (Clermont-Ferrand). Dans ces conditions, la cadence de reconstruction est de 7.9 à 8.3 $images/s$ (non temps-réel). Quelques images, ainsi que les caractéristiques des vidéos sont données sur la Figure 3.15 et les résultats de reconstructions sur les Figures 3.16 et 3.17. Les deux trajectoires sont quasiment bouclées, et on remarque que la dérive est minimale (de l'ordre de quelques mètres) par rapport à la distance parcourue. Les écarts moyens de positionnement par rapport à une solution obtenue par ajustement de faisceaux global (appliqué en fin du calcul incrémental) sont de 0.036% et 0.016% de la trajectoire totale soit environ 25.4 et 6.5 cm d'erreur.

(a) images tirées des vidéos. haut : Compiègne. bas : Clermont-Fd.

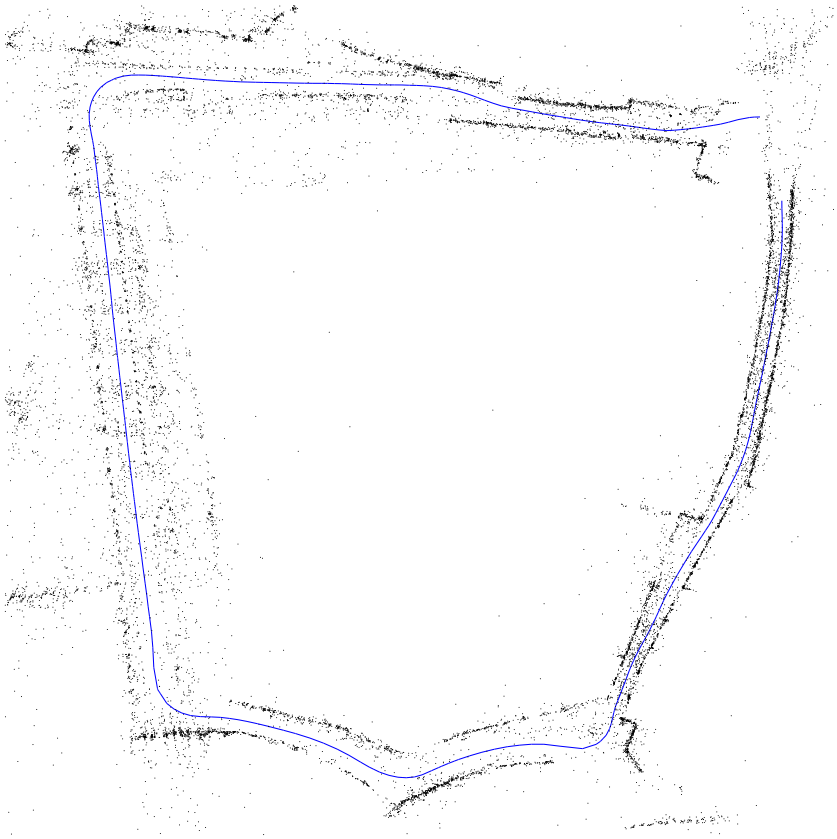
Séquence	#img	#img clef	#pts3D	#pts2D	Longueur (m)
Compiègne	2.900	185	10.395	43.226	700
Clermont-Fd	2.730	298	18.403	70.983	400

(b) caractéristiques techniques.

FIG. 3.15 – Séquences urbaines à Compiègne et Clermont-Ferrand.

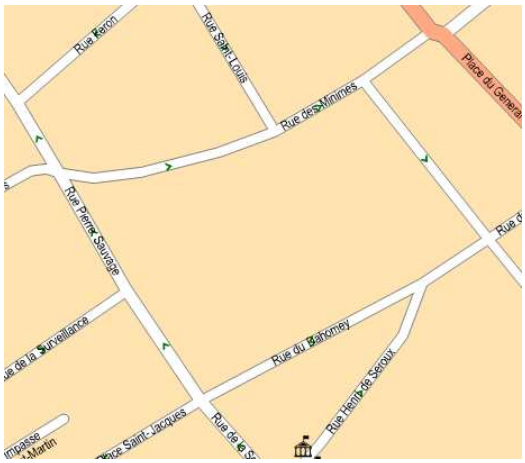


(a) carte routière du trajet emprunté.

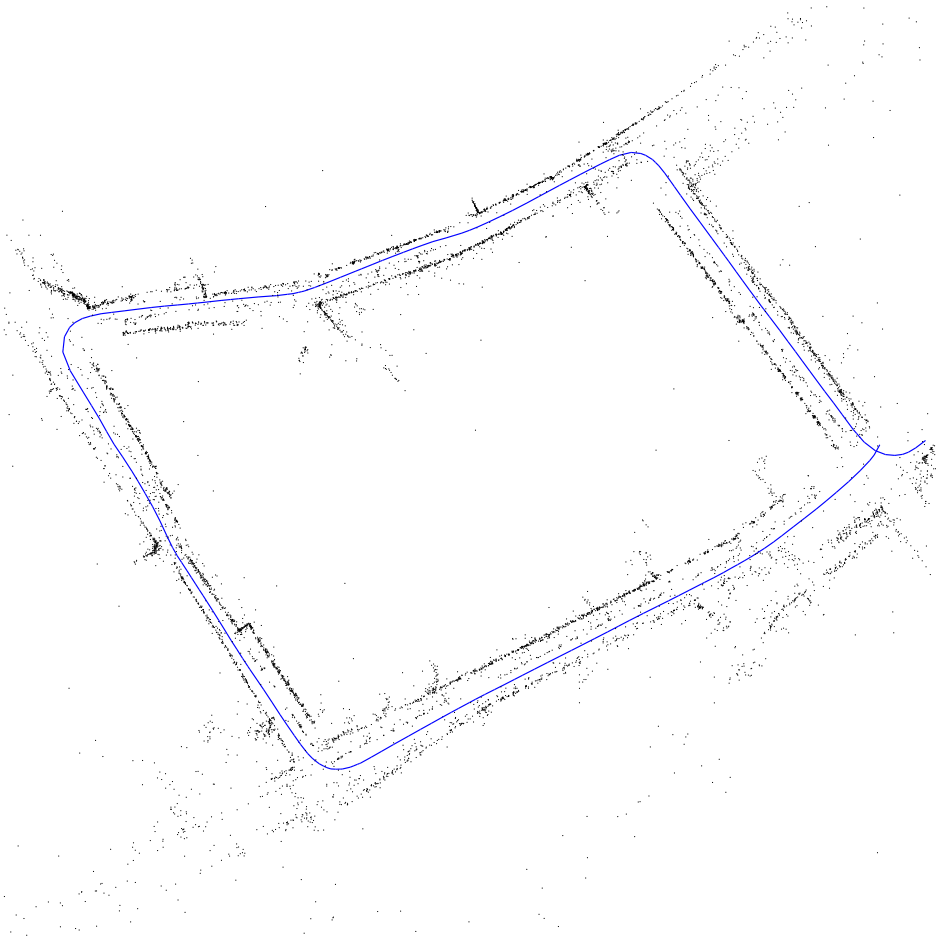


(b) résultat de la reconstruction 3D (vue de dessus).

FIG. 3.16 – Séquence "Clermont-Ferrand" : 298 poses de caméra, 18.403 points 3D.



(a) carte routière du trajet emprunté.



(b) résultat de la reconstruction 3D (vue de dessus).

FIG. 3.17 – Séquence "Compiègne" : 185 poses de caméra, 10.395 points 3D.

3.7.5 Influence du calibrage

Un des problèmes majeurs auxquels on est confronté lorsque l'on traite des longues séquences vidéo comme celles présentées dans la section précédente, est la dérive de la trajectoire calculée. Celle-ci s'explique en partie par le caractère incrémental de la méthode de reconstruction, qui peut entraîner une accumulation des erreurs. En effet, l'algorithme est récursif : les premières poses de caméra calculées sont utilisées pour la reconstruction des points 3D, qui sont ensuite eux-mêmes utilisés pour le calcul d'une nouvelle pose de caméra, et ainsi de suite. Une imprécision à un moment donné entraînera fatalement des erreurs dans la suite du processus. Une attention particulière doit être prise afin que chaque élément de la chaîne soit estimé le mieux possible. Pour cela, nous avons pris soin de calibrer précisément les caméras avant les reconstructions afin de limiter au maximum les problèmes liés aux paramètres intrinsèques des caméras. Les expériences précédentes ont montré que dans ce cas, la dérive est mineure. Cependant, pour des raisons techniques, ce calibrage précis peut ne pas être disponible. Qu'en est-il si la caméra qu'on utilise n'est que grossièrement calibrée ? C'est la question à laquelle on s'intéresse dans cette section.
Les deux principaux paramètres ; distance focale et point principal ont été testés. Pour cela on a utilisé une séquence de référence (séquence "Clermont-Ferrand") où la trajectoire est une boucle de 400 m presque fermée. Le résultat de reconstruction avec les paramètres de caméras déterminés lors de l'étape de calibrage est donné sur la Figure 3.16. Des modifications de $\pm 2.5\%$ et $\pm 5\%$ ont été appliquées à la focale $\mathbf{f}_0 = (f_x, f_y) = (264.4, 264.4)$ et au point principal $\mathbf{p}_0 = (u_0, v_0) = (258.4, 207.9)$. Cela représente des modifications de 6.6 pix et 13.2 pix pour la focale, de 8.2 pix et 16.5 pix pour le point principal. Les trajectoires obtenues avec ces modifications sont données sur la Figure 3.18(a). Pour mesurer l'écart avec la trajectoire de référence obtenue avec le bon calibrage, on a réalisé plusieurs mesures (voir Tableau 3.18(b), Figure 3.18). Tout d'abord, la longueur de la trajectoire permet de mesurer si il y a une réduction ou un agrandissement de la trajectoire (dérive dûe au facteur d'échelle). On a mesuré la trajectoire relative L/L_0 (L_0 étant la longueur de la trajectoire de référence) qui est égale à 1 si la longueur de la trajectoire n'est pas modifiée. Pour mesurer la dérive, on a également noté la distance entre le point de départ (situé en (0,0)) sur la Figure 3.18(a)) et le point d'arrivée, qui doit être proche de 0 puisque la trajectoire est quasiment bouclée. Il est également intéressant de voir la dérive suivant l'axe vertical. Finalement, on a mesuré la différence de cap à l'arrivée, par rapport à la

référence calibrée.

Influence de la distance focale

D'après les expériences, une sous-estimation de la focale entraîne un agrandissement de l'échelle au cours de la reconstruction, et au contraire on observe une réduction de l'échelle en cas de sur-estimation. Les distances sont fortement modifiées et cela produit des dérives importantes (jusqu'à plus de 200 m avec +5%) et des erreurs sur la longueur de trajectoire (jusqu'à 2.4 fois la longueur initiale). Les angles s'en trouvent également altérés avec une erreur de cap en fin de séquence d'au moins 8 degrés.

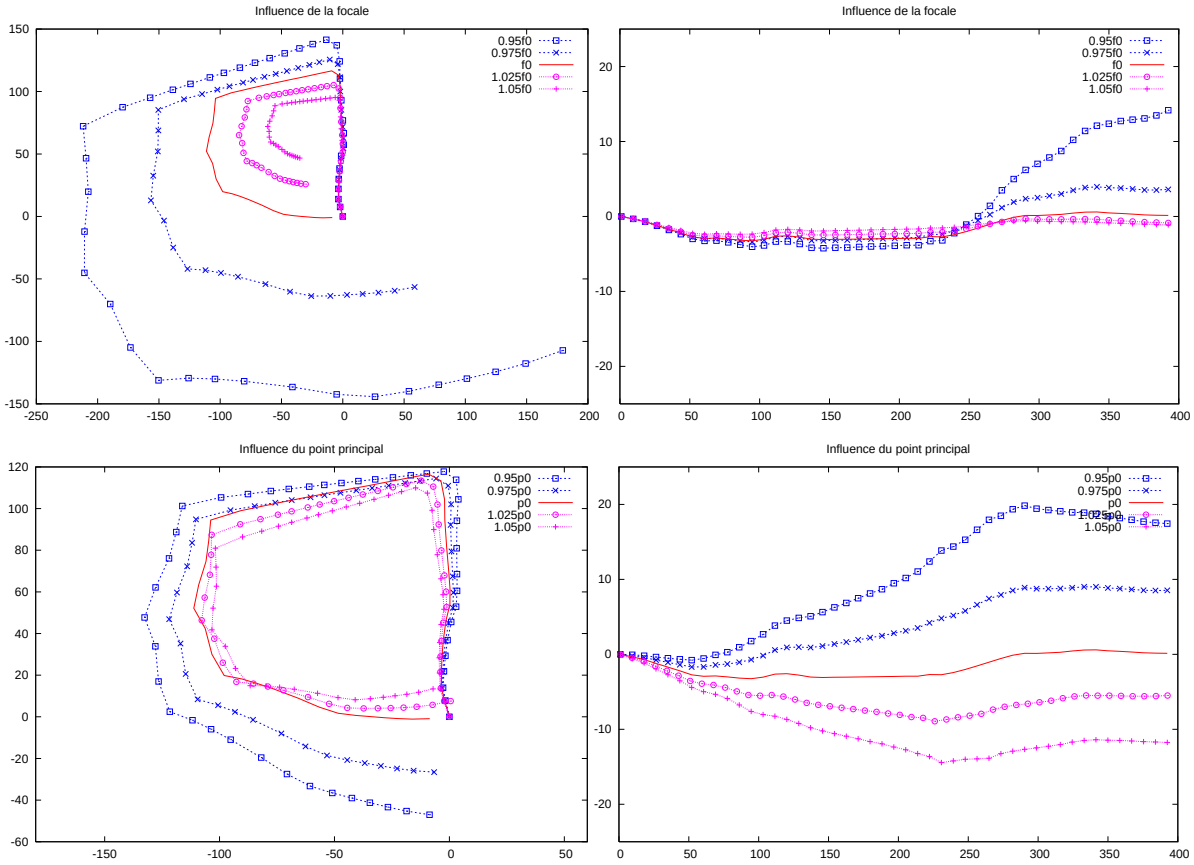
Influence du point principal

Les résultats sont différents en ce qui concerne l'influence du point principal. Les distances sont beaucoup moins modifiées (au plus 20% d'écart et une dérive maximale de 47 m) et ce sont surtout les angles qui sont influencés. On peut voir la dérive angulaire importante dès le début de la trajectoire au dessus du point (0,0) sur la Figure 3.18(a) (en bas à gauche). Cela entraîne une dérive plus forte en altitude.

Ces expériences ont montré que la méthode est sensible aux perturbations sur les paramètres intrinsèques. Si l'on souhaite obtenir une reconstruction précise, alors il est nécessaire d'étalonner précisément la caméra.

3.7.6 Influence du sous-échantillonnage

Un point important de l'algorithme est le choix des images clef qui consiste à faire le compromis entre un large mouvement de caméra, favorable aux calculs 3D, et un petit mouvement, plus favorable à la mise en correspondance des points d'intérêt. En jouant sur les paramètres impliqués dans la sélection et définis dans la section 3.4.2, notamment le paramètre M (nombre minimal de points en correspondance entre deux images clef), il est possible de rapprocher ou d'éloigner les images clef. Les expériences sur la séquence "Clermont-Ferrand" ont montré que le sous-échantillonnage a une influence faible sur le calcul de la trajectoire, comme illustré sur la Figure 3.19(a). En effet, pour cette séquence, l'écart de trajectoire en faisant varier M de 200 à 400 est toujours inférieur à 4 m dans le plan horizontal et 1 m en altitude. Cependant, l'écart entre les images clef agit sur le processus de reconstruction (voir Tableau 3.19(b), Figure 3.19). Par exemple, des images



(a) Influence du calibrage sur la trajectoire. haut : influence de la distance focale, bas : influence du point principal, gauche : dans le plan horizontal, droite : suivant l'axe vertical.

$\mathbf{f}$	L/L_0	dérive (m)	dérive alt (m)	$\gamma - \gamma_0$ (deg)
0.95 $\mathbf{f}_0$	2.40	229.41	15.06	17.66
0.975 $\mathbf{f}_0$	1.56	90.75	3.75	8.81
$\mathbf{f}_0$	1.00	1.11	0.14	0.00
1.025 $\mathbf{f}_0$	0.72	36.95	-0.91	-8.16
1.05 $\mathbf{f}_0$	0.52	57.18	-1.17	-15.78

$\mathbf{p}$	L/L_0	dérive (m)	dérive alt (m)	$\gamma - \gamma_0$ (deg)
0.95 $\mathbf{p}_0$	1.20	47.83	17.36	-11.32
0.975 $\mathbf{p}_0$	1.09	26.54	8.57	-6.09
$\mathbf{p}_0$	1.00	1.11	0.14	0.00
1.025 $\mathbf{p}_0$	0.96	7.60	-5.50	6.37
1.05 $\mathbf{p}_0$	0.92	15.61	-11.79	12.76

(b) Longueur de trajectoire, dérive 3D, dérive en altitude, et dérive de l'orientation.

FIG. 3.18 – Influence du calibrage.

clef plus nombreuses et rapprochées vont permettre de reconstruire un nuage de points plus dense. C'est ce qu'illustre la Figure 3.19(c). Pour $M = 200$, on obtient 172 images clef (soit environ 1 image sur 16) et un nuage de 7.105 points 3D alors que pour $M = 400$ on obtient 497 images clef (1 image sur 5) et 39.810 points 3D. En revanche, la reconstruction d'un plus grand nombre de points se fait au détriment d'un temps de calcul supérieur et une vitesse de reconstruction moyenne qui passe de 11.1 $images/s$ à 6.1 $images/s$. On remarque également que l'erreur de reprojection a tendance à augmenter en rapprochant les images clef.

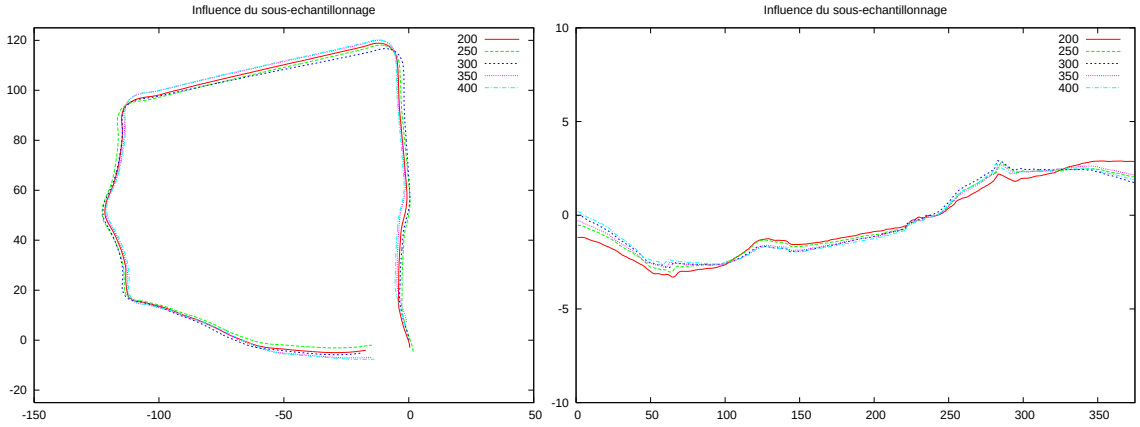
Selon l'application, on pourra donc être amené à choisir entre une grande vitesse de reconstruction si seule la trajectoire est importante ou alors une reconstruction moins rapide si un modèle 3D plus dense est requis.

3.8 Applications de la méthode de reconstruction 3D incrémentale

La méthode de reconstruction incrémentale présentée est assez générale. On peut considérer le système caméra + ordinateur comme un capteur fournissant une position (ou une trajectoire) et des informations tridimensionnelles sur l'environnement (sous la forme d'un nuage de points 3D). L'application la plus directe concerne l'odométrie pour l'automobile mais on peut envisager de décliner cette méthode vers un ensemble d'applications variées. Comme on va le voir dans cette section, plusieurs de ces applications ont déjà été étudiées au cours de ces travaux de thèse, de projets ou de stages en parallèle mais d'autres sont encore à explorer.

3.8.1 Odométrie visuelle temps-réel pour l'automobile en milieu urbain

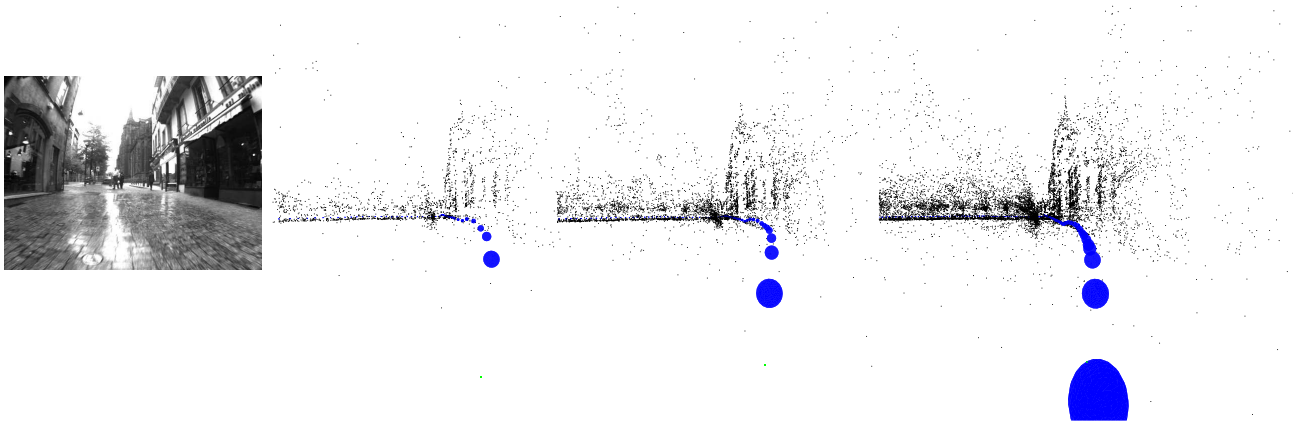
Dans le cadre du projet ODIAAC (ODométrie Intégrée pour des Applications d'Aide à la Conduite en milieu urbain), le LASMEA et le CEA LIST collaborent dans le développement d'outils permettant la localisation d'un véhicule dans son environnement. Un des enjeux est la fusion des informations fournies par un récepteur GPS avec les informations d'odométrie visuelle, afin de pallier les déficiences du système de positionnement global en cas de perturbations du signal satellite (bâtiments, arbres, etc...). Le stage de A.



(a) Influence sur la trajectoire. gauche : dans le plan horizontal, droite : suivant l'axe vertical.

M	#img clef	#pts 3D	#pts 2D	Vitesse (img/s)	RMS (pix)
200	172($\simeq$1/16)	7.105	25.062	11.1	0.55
250	228($\simeq$1/12)	11.760	43.492	9.7	0.63
300	298($\simeq$1/9)	18.403	70.983	8.6	0.54
350	385($\simeq$1/7)	27.120	110.572	7.1	0.59
400	497($\simeq$1/5)	39.810	169.576	6.1	0.72

(b) Influence sur la reconstruction 3D : nombre d'images clef, nombre de points 3D/2D, vitesse de calcul et RMS.



(c) Densité du nuage de points 3D. Reconstruction de la cathédrale de Clermont-Ferrand pour $M = 200$, $M = 300$, $M = 400$. Vue perspective du nuage de points 3D et de la trajectoire par une image clef.

FIG. 3.19 – Influence du sous-échantillonnage.

Eudes [18] a consisté à valider la possibilité d'utiliser notre méthode d'odométrie visuelle dans un cadre urbain, où les véhicules se déplacent jusqu'à 50 km/h. Les problèmes liés à la vitesse du véhicule viennent de la distance plus importante parcourue entre deux images et donc de la perte plus rapide des points servant à la localisation. A cette vitesse et pour une fréquence d'acquisition de caméra de 15 $images/s$, la distance parcourue est proche de 1 m entre deux images successives.
Des améliorations ont dû être apportées afin de satisfaire la contrainte temps-réel avec ces paramètres.

Utilisation d'un descripteur plus discriminant

Un nouveau descripteur a été choisi. Il s'agit du descripteur récent SURF [4] qui est plus rapide à calculer que SIFT [51] et plus discriminant que la corrélation, et permet de ce fait de réduire le risque d'erreur dans les appariements de points entre deux images, et donc d'avoir de meilleurs points pour la phase de reconstruction. Cette constatation amène à réduire le nombre de points utilisés dans les appariements, ce qui permet notamment de diminuer le temps de calcul des descripteurs.
SURF étant plus discriminant, il permet d'agrandir fortement les zones de recherche sans perte de performance sur la mise en correspondance. On peut ainsi utiliser de grandes zones de recherche (80×50 pix) et être capable de suivre les points se déplaçant rapidement sur de plus longues distances, par exemple lors des virages.
L'utilisation de ce nouveau descripteur permet aussi de réduire le nombre d'images clef qui sont coûteuses en temps de calcul. En suivant des points qui ont des mouvements beaucoup plus importants, on peut choisir des images clef plus éloignées les unes des autres. SURF permet également d'apparier les points entre deux images clef aussi bien que la corrélation avec la méthode indirecte, et ainsi de supprimer des images intermédiaires. Grâce à cela, on a pu valider la méthode sur des tests synthétiques où l'accélération virtuelle du véhicule est réalisée par un sous-échantillonnage du flux d'images, ainsi que sur des séquences réelles à 50 km/h.

Optimisation de l'implémentation logicielle

La tendance actuelle au niveau des microprocesseurs est d'augmenter le nombre de *cores* (unités de traitement) plutôt que d'augmenter la fréquence de fonctionnement. De manière à profiter de toute la puissance des nouveaux

processeurs, le calcul des points d'intérêt et des descripteurs a été séparé du reste du programme. La parallélisation diminue considérablement le temps de calcul et l'utilisation d'un deuxième *core* permet d'utiliser SURF dans un temps similaire à la corrélation non parallélisé. Cependant, cette parallélisation du code augmente la complexité et les difficultés de débogage car il est nécessaire de gérer la non-simultanéité des tâches.
Pour aller encore plus loin dans l'optimisation et le calcul parallèle, le GPU (Graphical Processeur Unit) de la carte graphique a été utilisé pour calculer les points d'intérêt. Cela permet d'alléger la tâche du processeur. La nouvelle version GPU du détecteur de HARRIS est environ 12 fois plus rapide que l'algorithme de base.

La Figure 3.20 montre le résultat de l'odométrie visuelle traitée en temps-réel avec un véhicule roulant à 50 km/h. La longueur de la trajectoire est d'environ 2 km (avec demi-tour) et la dérive est assez importante.

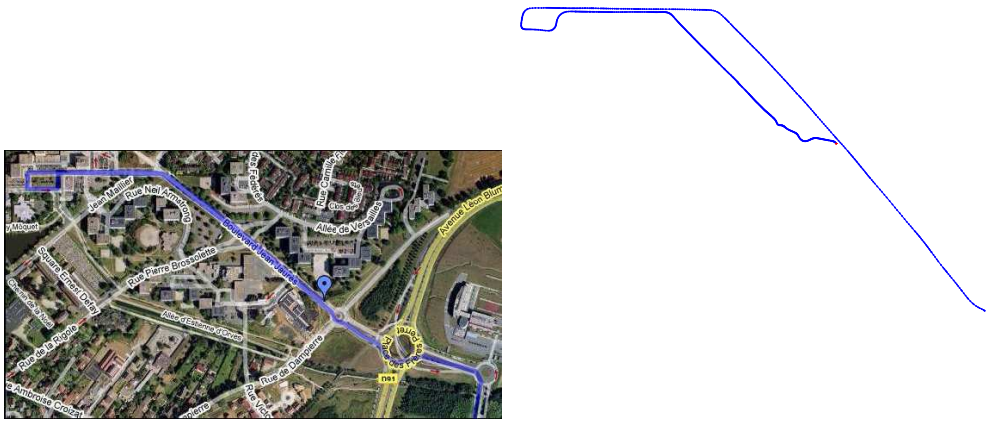


FIG. 3.20 – Vue par satellite et odométrie visuelle de la trajectoire.

3.8.2 Localisation d'un robot mobile ou "véhicule intelligent"

Dans le cadre de la localisation par vision d'un robot mobile autonome (Figure 3.21), il est possible d'utiliser une carte tridimensionnelle. Dans les travaux de thèse de E. ROYER [83, 82], cette carte contient une représentation par nuage de points de l'environnement dans lequel évolue le robot, ainsi que la trajectoire précise à emprunter. Elle est générée à partir d'une

séquence vidéo de référence réalisée avec une caméra embarquée sur un véhicule guidé manuellement. Cette première étape correspond à une phase de cartographie. Le but est ensuite pour le robot de parcourir à nouveau la même trajectoire lors d'une seconde phase de navigation automatique. La localisation du robot est alors obtenue en temps-réel, en comparant les images en mémoire (associées à la structure 3D calculée) avec les images courantes fournies par la même caméra.
La cartographie de référence utilisée par Royer *et al.* est calculée par ajustement de faisceaux hiérarchique. Elle a été remplacée avec succès dans cette étude par la reconstruction obtenue avec notre méthode incrémentale. Les images correspondant à cette expérience sont présentées sur la Figure 3.22(a). L'avantage de la nouvelle méthode est la très grande diminution du temps de calcul nécessaire (qui passe de plusieurs heures à quelques minutes) pour l'obtention de la reconstruction 3D de référence. De plus, la méthode de reconstruction initiale limite le nombre d'images de la séquence à traiter alors que notre algorithme lève cette contrainte.

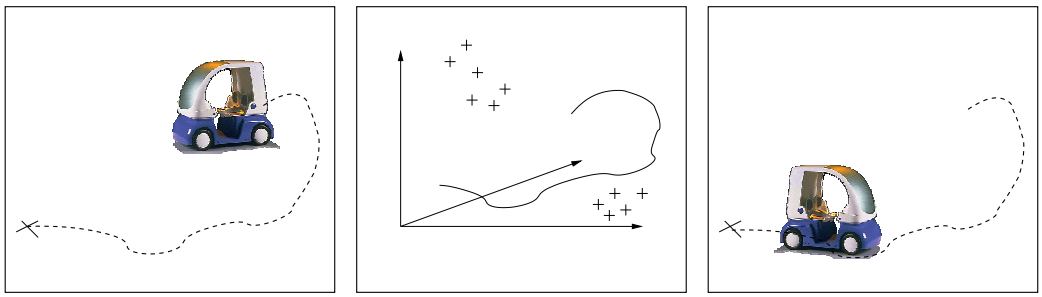
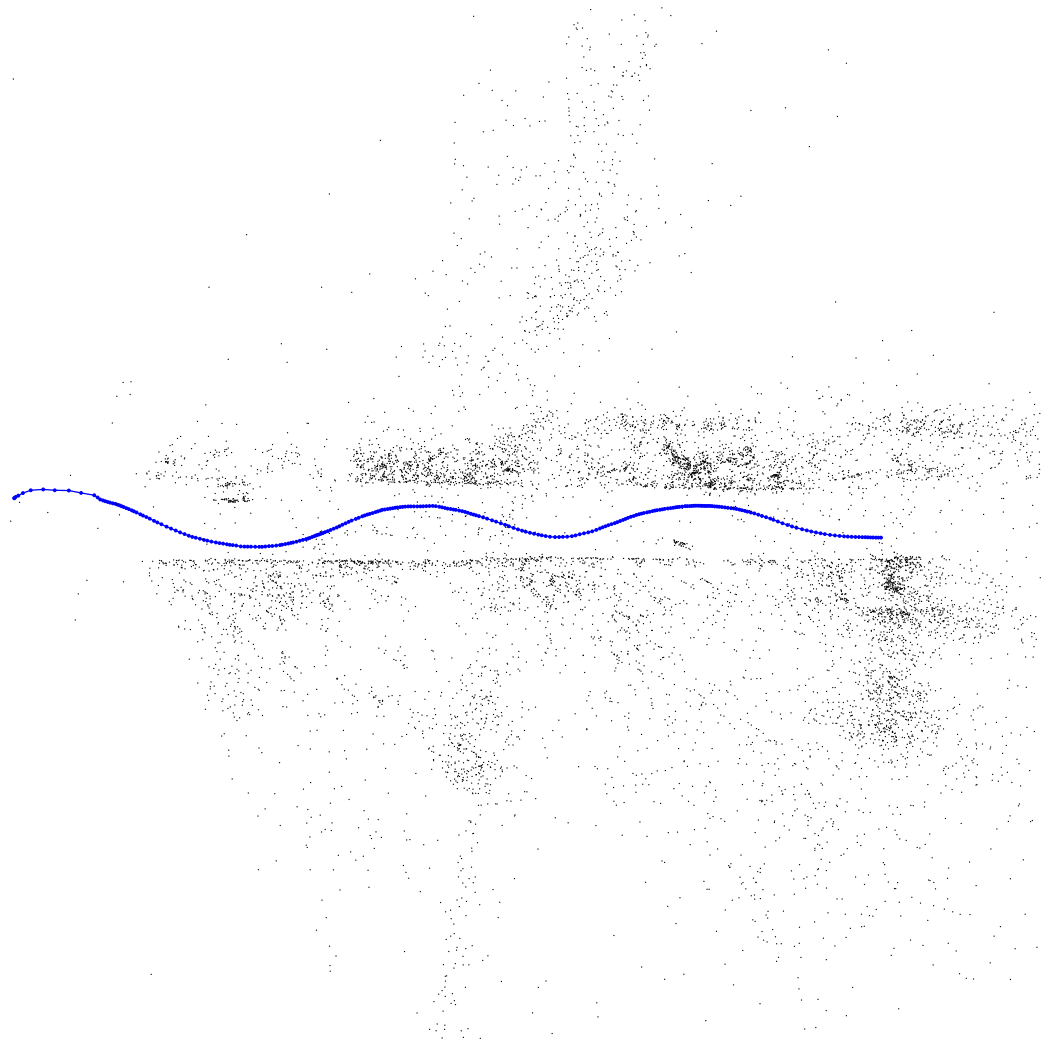


Fig. 3.21 – Principe de la localisation par vision. 1) le véhicule navigue librement. 2) on reconstruit la trajectoire ainsi que des amers visuels. 3) le véhicule peut se localiser et rejouer la même trajectoire.

Perspectives Ces résultats nous permettent de viser l'application de convoi automatique de véhicules intelligents par vision. L'idée est la suivante : le premier véhicule est toujours conduit manuellement. Au cours de son déplacement, la reconstruction de référence est calculée et la trajectoire à suivre est transmise aux autres véhicules. Grâce à la reconstruction incrémentale, les véhicules qui suivent peuvent partir sans attendre la fin du déplacement du véhicule de tête, et les données de référence sont mises à jour et re-transmises régulièrement.

(a) 3 images tirées de la séquence.



(b) Vue de dessus de la reconstruction 3D calculée et utilisée comme cartographie de référence pour la localisation du robot mobile.

FIG. 3.22 – Séquence "localisation" : images et reconstruction 3D.

3.8.3 Localisation de véhicule pour le stationnement automatique

Lors du stage de L. Leyrit [42] au LASMEA, nous nous sommes intéressés au stationnement de véhicules de tourisme en milieu urbain. Ces travaux s'inscrivent dans les activités *localisation de véhicules par vision* du laboratoire et se sont déroulés en partenariat avec l'industrie automobile. L'objectif est de développer un outil de "créneau automatique" pour aider les conducteurs dans ces manoeuvres délicates, à l'aide de caméras installées dans les logements de phares des voitures.
Le système doit fonctionner de la façon suivante, en deux étapes :

- Tout d'abord, une phase de repérage. Lorsque le conducteur aperçoit une place libre, il enclenche le système de créneau automatique. Ensuite, il conduit sa voiture, en marche avant, le long de l'emplacement. Pendant ce temps, le système appréhende l'environnement extérieur et situe le véhicule dans ce contexte.
- A l'issue de cette première étape, le système prend en charge la direction de la voiture pour la garer dans l'emplacement repéré. L'automobiliste ne se charge que de la vitesse de son véhicule et le créneau est exécuté en mode automatique.

Lors de la première étape en marche avant, cette localisation doit permettre au système de repérer l'emplacement vide et de situer la position de la voiture par rapport à celle-ci. Lorsque le système de commande guidera en marche arrière la voiture, les caméras devront permettre de contrôler la position effective du véhicule.
Deux pistes ont été explorées. La première consistait à réaliser une reconstruction de l'environnement lors de la marche avant, et ensuite une simple localisation de la caméra (et ainsi du véhicule) dans cet environnement reconstruit, de la même manière que dans le paragraphe précédent. L'avantage de cette solution est qu'une reconstruction 3D très rapide n'était pas nécessaire, puisque le véhicule peut attendre quelques secondes à la fin de sa marche avant. Malheureusement, cette méthode n'est pas applicable dans ce contexte car le changement de point de vue entre la marche avant et la marche arrière est trop important, à cause de la grande rotation du véhicule. De ce fait, les points 3D reconstruits ne sont plus visibles en marche arrière et la localisation est impossible.
La deuxième solution, celle finalement retenue, était de poursuivre la reconstruction 3D lors de la manoeuvre complète. Ainsi, même pendant la marche arrière, on continue à cartographier l'environnement et à localiser le véhicule

en même temps. Pour cela, une méthode rapide de reconstruction était nécessaire et on a choisi la méthode de reconstruction incrémentale présentée dans ce chapitre. Dès la fin de l'initialisation, la méthode fournit la localisation du véhicule dans le repère de référence, et le nuage de points 3D est complété à chaque image clef. Celui-ci sera utilisé pour localiser automatiquement la place de stationnement libre.

Spécificités de l'application

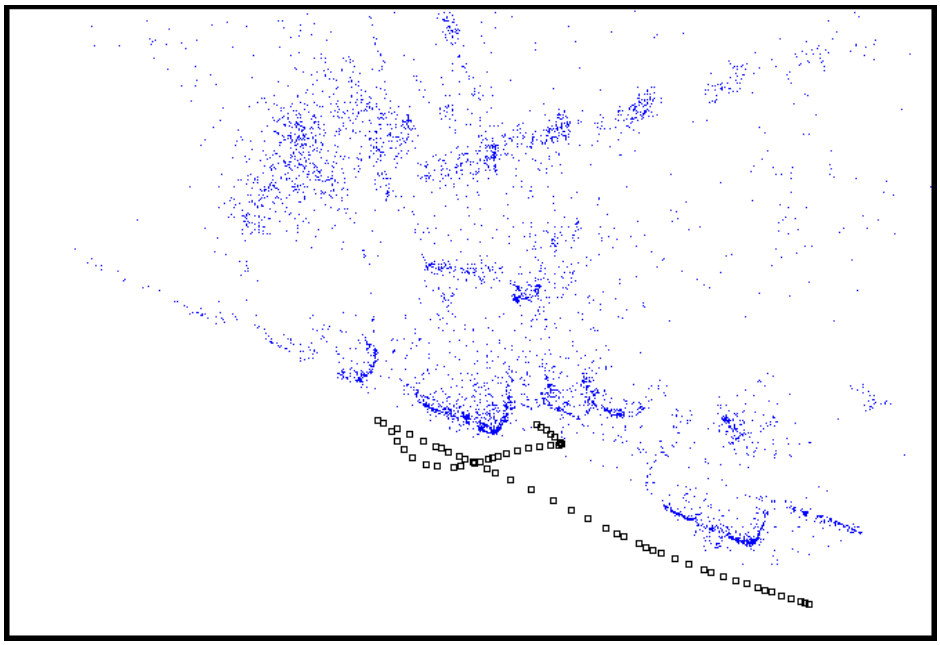
Le projet de stationnement automatique vise une application très particulière. La géométrie de la scène (véhicules stationnés), la position de la caméra (illustrée sur la Figure 3.23(c)) et son mouvement (latéral) ont conduit à plusieurs adaptations.
Les principaux points étudiés ont été :

1. Modification de la taille et de l'emplacement des fenêtres de recherche pour l'appariement des points. Pour apparier un point de l'image 0 à un point de l'image 1, la fenêtre de recherche dans l'image 1 est centrée sur l'emplacement du point dans l'image 0 avec un rapport largeur/hauteur d'environ 4/3. Étant donné la nature du mouvement, les points se déplacent presque horizontalement, toujours dans le sens opposé au mouvement. On a élargi les fenêtres et diminué leur hauteur pour atteindre un rapport 5/1.
2. Prise en compte de la distorsion pour l'appariement des points. A cause des effets de distorsion, les points ne se déplacent pas tout à fait horizontalement mais suivant des lignes courbes. L'emplacement des fenêtres est choisi à partir des points corrigés de la distorsion.
3. Calcul du gabarit. Le calcul des points 3D obtenus à la fin de la marche avant a été exploité pour la détermination de l'emplacement de la "place libre" (Figures 3.23(b) et 3.23(d)). Celle-ci est basée sur un filtrage grossier des points, et une recherche des droites qui passent au mieux par les projections des points au sol.

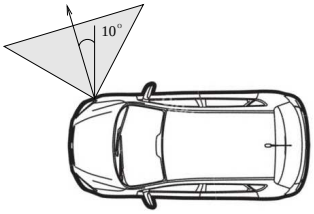
3.8.4 Calibrage des systèmes multi-caméras

Une application plus inattendue concerne le calibrage d'un ensemble rigide de caméras. Considérons n caméras C^1, C^2, ..., C^n rigidement liées les unes par rapport aux autres. Par exemple, on peut retrouver cette configuration dans l'application automobile précédente (une caméra dans chaque

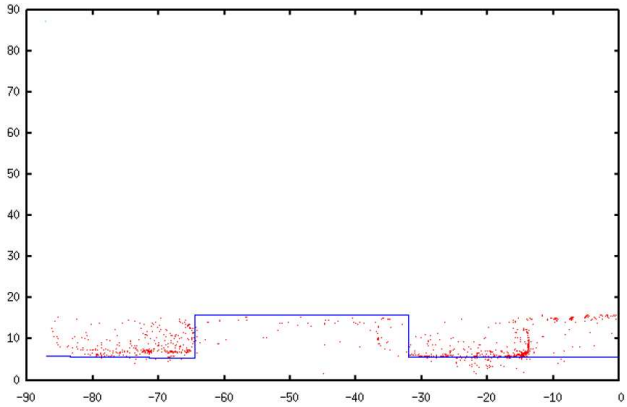
(a) 3 images tirées de la séquence.



(b) Vue de dessus de la reconstruction 3D.



(c) Position de la caméra.



(d) Résultat : calcul du gabarit (la courbe délimite l'emplacement libre).

FIG. 3.23 – Manoeuvre de créneau. (a) images, (b) reconstruction 3D, (c) position de la caméra et (d) calcul du gabarit.

phare) ou dans les systèmes de vidéo-surveillance. De tels systèmes seront évoqués au chapitre suivant sur les caméras génériques. Le but de cette application est de déterminer les positions $(\mathtt{R}^1, \mathbf{t}^1), (\mathtt{R}^2, \mathbf{t}^2), ..., (\mathtt{R}^n, \mathbf{t}^n)$ des n caméras dans un repère fixe. Les paramètres intrinsèques de toutes les caméras sont supposés connus.

L'idée développée est la suivante. On utilise une caméra mobile supplémentaire qui se déplace à proximité des caméras du système à calibrer, et on réalise une reconstruction 3D. Pendant son mouvement, on veille à ce que la caméra mobile passe dans une position proche de chaque caméra fixe, de manière à obtenir des champs de vue rapprochés. Cela permet de mettre en correspondance chaque image provenant d'une caméra fixe C^i avec une image clef de la séquence reconstruite et d'accéder ainsi à la pose $(\mathtt{R}^i, \mathbf{t}^i)$ dans le repère de la reconstruction. Le choix de l'image clef la plus proche d'une caméra fixe est automatique : l'algorithme met en correspondance l'image fournie avec l'ensemble des images clef et choisit celle qui donne le plus grand nombre de points d'intérêt appariés.

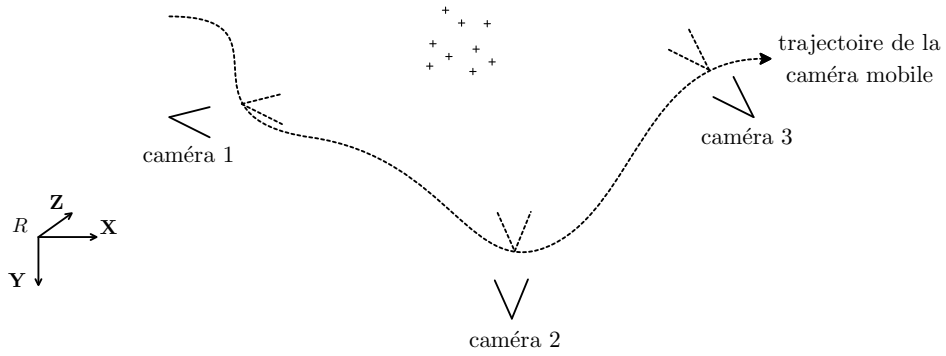
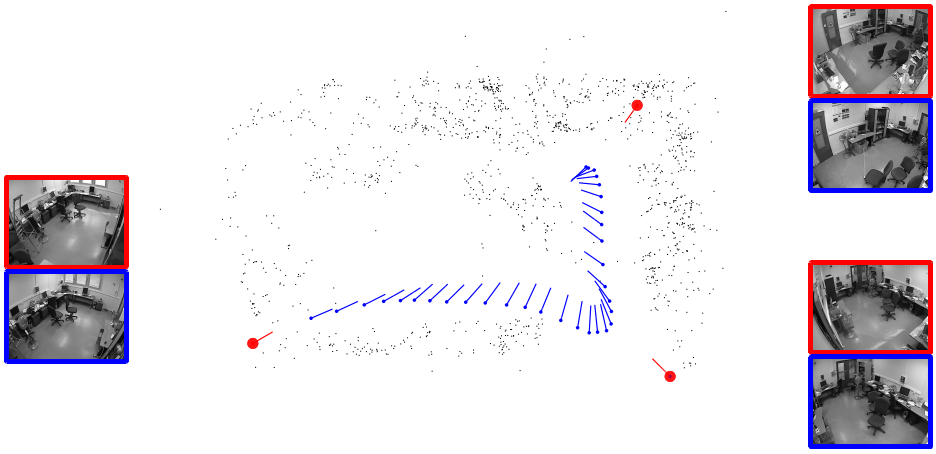


FIG. 3.24 – Principe du calibrage multi-caméras.

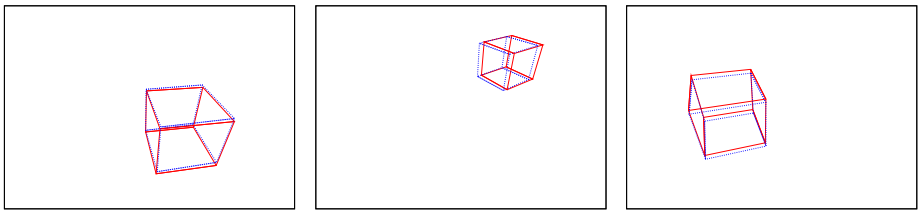
Dans le cas particulier où toutes les caméras ont un champ de vue commun, cette méthode est sans aucun doute moins précise que les approches classiques utilisant une mire de calibrage visible dans toutes les images. Ici, les positions relatives des caméras ne sont pas calculées à partir des observations d'un objet 3D parfaitement connu mais à partir de points de la scène reconstruits d'après le mouvement de la caméra mobile. De plus, le calibrage est obtenu à un facteur d'échelle près.
Les premiers résultats obtenus sur un système de vidéo-surveillance composé de trois caméras sont prometteurs (Figure 3.25). La comparaison après mise à l'échelle avec le calibrage de référence calculé à partir de cibles carrées de 40 *cm* (visibles sur la Figure 3.25(b)) donne une erreur de l'ordre de 3 *cm*

sur la position des caméras et 2° sur l'orientation.



(a) Vue de dessus de la reconstruction 3D et du système multi-caméras. Images fournies par les caméras fixes et images clef correspondantes.

(b) Images des cibles utilisées pour le calibrage de référence.



(c) Projection d'un cube virtuel dans les trois caméras : $5.2 < p - p_{ref} < 9.9\ pix.$

FIG. 3.25 – Résultats du calibrage multi-caméras

Dans un cadre plus général, cette nouvelle méthode est très intéressante car elle offre la possibilité de calibrer des systèmes de caméras qui ont des champs de vue complètement disjoints, ce qui n'est pas possible avec les algorithmes classiques. De plus, la mise en oeuvre est assez simple et il est possible d'utiliser une simple caméra portée à la main.

Chapitre 4

Reconstruction 3D incrémentale par ajustement de faisceaux local : généralisation aux caméras génériques

Dans ce dernier chapitre, on explique la généralisation de la méthode de reconstruction incrémentale. Elle rend possible l'utilisation d'autres types de caméras mais implique des modifications dans plusieurs étapes de l'algorithme. En effet, le calcul initial de la géométrie ainsi que l'ajustement de faisceaux local et le calcul de pose, initialement dédiés aux caméras perspectives, doivent être revisités. La méthode générique complète a été publiée dans [66].

4.1 Introduction aux caméras génériques

Si l'on souhaite utiliser différents types de caméras pour la reconstruction 3D, alors on peut adapter la méthode présentée à d'autres modèles de caméras. Plusieurs modèles ont été étudiés et utilisés : perspectifs, fish-eyes, catadioptriques, systèmes de multi-caméras (par exemple : une paire stéréo rigide), etc. Beaucoup d'algorithmes spécifiques à un modèle de caméra donné ont été développés avec succès et sont maintenant couramment utilisés pour les modèles perspectifs et stéréo [73, 70, 45]. Les cas des caméras omnidirectionnelles centrales (catadioptriques et fish-eye) ou non centrales (multi-caméras) qui offrent un grand champ de vue ont aussi été exami-

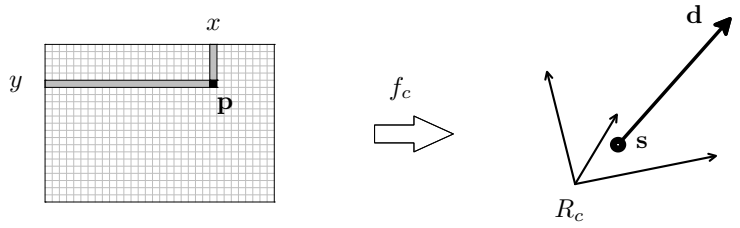
nés [7, 58, 71]. Cependant, l'utilisation de ces algorithmes spécifiques devient délicate si l'on souhaite inter-changer ou combiner différentes caméras. Développer des outils de reconstruction 3D génériques exploitables pour tout type de caméra reste un défi technique intéressant. En 2003, PLESS [71] a proposé de considérer un ensemble rigide de plusieurs caméras comme une seule et unique caméra en introduisant la notion de contrainte épipolaire généralisée. D'autres auteurs ont examiné le problème du calibrage en introduisant également des modèles de caméras génériques [98, 34]. Dans un modèle de caméra générique, chaque pixel définit un rayon dans le système de coordonnées des caméras qui peut intersecter ou non un point unique appelé centre de projection. Dans les travaux récents sur le SfM (Structure from Motion) générique, le mouvement de caméra peut être estimé à l'aide de la matrice essentielle généralisée [71, 77] donnée par l'équation de PLESS [71] ou avec des méthodes minimales d'estimation de pose relative [96].

De la même manière que pour les modèles spécifiques, une méthode est nécessaire pour le raffinement des points 3D et des poses de la caméra. Dans le cas générique où plusieurs types de caméras sont possibles (perspective, stéréo, catadioptrique ...), l'ajustement de faisceaux est différent de la méthode classique utilisée pour les caméras perspectives où l'on minimise une erreur de reprojection dans les images. L'erreur minimisée peut être une erreur 3D ou 2D [77]. Puisque la fonction de projection n'est pas explicite pour certains modèles de caméras, on peut faire le choix d'utiliser une erreur 3D [58]. Une telle erreur n'est pourtant pas optimale car elle favorise la contribution des points éloignés des caméras par rapport aux points proches, et cela peut produire des résultats biaisés [33, 52]. Une autre solution est de minimiser une erreur de reprojection 2D (en pixels) en regroupant les rayons dans des faisceaux, chaque faisceau approximant une caméra centrale [77]. Notre méthode générique est basée sur les rayons s'échappant de la caméra et minimise une erreur d'angle entre des rayons. Naturellement, le premier avantage est la possibilité d'échanger facilement un modèle de caméra par un autre. Le second avantage est que la méthode reste efficace même si la fonction de projection n'est pas explicite (comme dans le cas catadioptrique non-central).

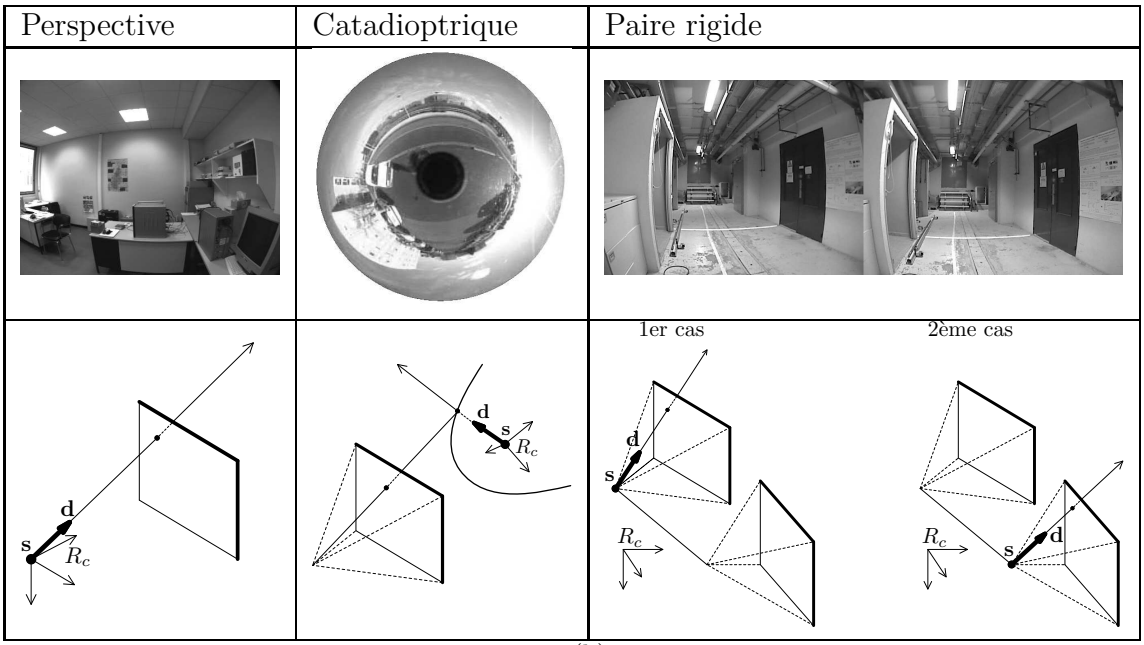
4.2 Le modèle de caméra générique

Quel que soit le type de caméra utilisé (y compris un système rigide de plusieurs caméras), on considère que la caméra fournit une seule et même

image (composition d'images dans le cas multi-caméra). Pour tout pixel $\mathbf{p}$ de l'image générique, la fonction de calibrage f_c (connue) de la caméra définit un unique rayon optique $r = f_c(\mathbf{p})$. Ce rayon est une droite orientée $r = (\mathbf{s}, \mathbf{d})$ avec $\mathbf{s}$ le point de départ ou origine et $\mathbf{d}$ la direction du rayon dans le repère de la caméra (avec $\|\mathbf{d}\| = 1$). Pour une caméra centrale, $\mathbf{s}$ est un point unique (le centre) pour tout pixel $\mathbf{p}$. Dans le cas général, $\mathbf{s}$ peut être n'importe quel point donné par f_c (Figure 4.1).



(a)



(b)

FIG. 4.1 – Le modèle générique : (a) correspondance pixel→rayon, (b) 3 exemples d'images génériques et rayons associés.

4.3 Raffiner la géométrie

4.3.1 Erreur générique

De manière générale, le raffinement de la géométrie consiste à faire correspondre au mieux le modèle calculé aux données mesurées dans les images. Pour cela, on minimise une somme de carrés $\sum_{i,j} ||\epsilon_j^i||^2$ où ϵ_j^i décrit l'erreur obtenue pour l'estimation du j-ème point et la i-ème caméra par rapport à la mesure correspondante.

L'approche classique [102, 65] consiste à prendre ϵ_j^i comme l'erreur de reprojection en pixels du j-ème point dans la i-ème image :

$$||\epsilon_j^i|| = dist\left(\mathbf{p}_j^i, f^i(\mathbf{P}_j)\right). \tag{4.1}$$

Puisque la fonction de projection f n'est pas la même suivant le type de caméra, l'erreur ainsi calculée est une erreur spécifique dépendante du modèle utilisé. Dans notre cas, il faut minimiser une erreur générique.

Soit $\bar{\mathbf{P}}_j = [X_j\ Y_j\ Z_j\ T_j]^\top$ les coordonnées homogènes du j-ème point $\mathbf{P}_j$ dans le repère global. Soient $\mathtt{R}^i$ et $\mathbf{t}^i$ l'orientation (matrice rotation) et la position de l'origine du repère de la i-ème caméra dans le même repère global.
Si $(\mathbf{s}_j^i, \mathbf{d}_j^i)$ est le rayon optique correspondant à l'observation du point $\mathbf{P}_j$ par la i-ème caméra, la direction de la ligne définie par $\mathbf{s}_j^i$ et $\mathbf{P}_j$ est

$$\mathbf{D}_j^i = {\mathtt{R}^i}^\top[\mathtt{I}_3 \mid -\mathbf{t}^i]\bar{\mathbf{P}}_j - T_j\mathbf{s}_j^i \tag{4.2}$$

dans le repère de la i-ème caméra.
Nous définissons ϵ_j^i tel que $||\epsilon_j^i||$ est l'angle entre les directions $\mathbf{d}_j^i$ et $\mathbf{D}_j^i$ définies précédemment (voir Figure 4.2) :

$$||\epsilon_j^i||_{(générique)} = angle\left(\mathbf{d}_j^i, \mathbf{D}_j^i\right). \tag{4.3}$$

Dans le cas idéal, les directions $\mathbf{d}_j^i$ et $\mathbf{D}_j^i$ sont parallèles (ce qui est équivalent à une erreur de reprojection de zéro pixel).

4.3.2 Choix de l'erreur angulaire

A première vue, on choisit ϵ_j^i comme la valeur exacte de l'angle formé par les vecteurs $\mathbf{d}_j^i$ et $\mathbf{D}_j^i$. Celle-ci s'exprime de la manière suivante :

$$\epsilon_j^i = \arccos(\mathbf{d}_j^i.\frac{\mathbf{D}_j^i}{||\mathbf{D}_j^i||}) \tag{4.4}$$

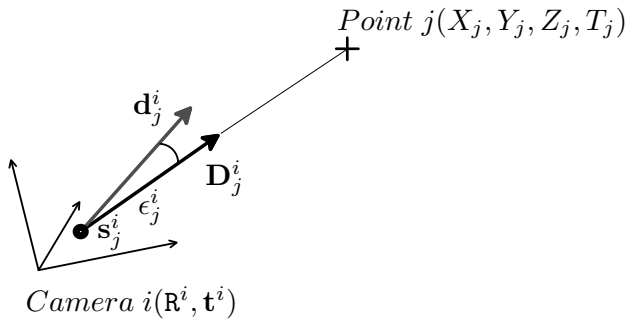


FIG. 4.2 – Angle entre le rayon correspondant à l'observation $(\mathbf{s}_j^i, \mathbf{d}_j^i)$ et le rayon passant par $\mathbf{s}_j^i$ et le point 3D $\mathbf{P}_j$. L'ajustement de faisceaux générique minimise une somme de carrés de tels angles.

puisque $\mathbf{d}_j^i \cdot \mathbf{D}_j^i = \left\|\mathbf{d}_j^i\right\| \left\|\mathbf{D}_j^i\right\| \cos\left(angle(\mathbf{d}_j^i, \mathbf{D}_j^i)\right)$ et $||\mathbf{d}_j^i|| = 1$. Cependant, des expériences ont montré que la méthode de LEVENBERG-MARQUARDT impliquée dans l'ajustement de faisceaux convergeait mal avec cette erreur. Une explication théorique à ceci est donnée dans [44]. Cette fonction d'erreur n'est pas continue $\mathcal{C}^1$ si ϵ_j^i=0 et c'est cette discontinuité à la solution exacte qui peut entraîner des problèmes de convergence.

Le choix de ϵ_j^i comme on le décrit maintenant s'est montré satisfaisant. Nous choisissons

$$\epsilon_j^i = \pi(\mathtt{R}_j^i \mathbf{D}_j^i) = \pi\left(\mathtt{R}_j^i \left(\mathtt{R}^{i\top}[\mathtt{I}_3 \mid -\mathbf{t}^i]\bar{\mathbf{P}}_j - T_j \mathbf{s}_j^i\right)\right) \tag{4.5}$$

où $\mathtt{R}_j^i$ est une matrice rotation telle que $\mathtt{R}_j^i \mathbf{d}_j^i = [0\ 0\ 1]^\top$ et $\pi\left([x\ y\ z]^\top\right) = [x/z\ \ y/z]^\top$. On note que ϵ_j^i est un vecteur 2D dont la norme euclidienne $||\epsilon_j^i||$ est égale à la tangente de l'angle entre $\mathbf{d}_j^i$ and $\mathbf{D}_j^i$. En effet, puisque $\left\|\pi\left([x\ y\ z]^\top\right)\right\| = \sqrt{\frac{x^2+y^2}{z^2}}$ est la tangente de l'angle formé par les vecteurs $[0\ 0\ 1]^\top$ et $[x\ y\ z]^\top$, on voit que $||\epsilon_j^i||^2 = \tan^2\left(angle(\mathbf{d}_j^i, \mathbf{D}_j^i)\right)$. La tangente est une bonne approximation de l'angle si celui-ci est petit, et c'est le cas dans ce contexte.

De la même façon que pour la méthode spécifique aux caméras perspectives, un ajustement de faisceaux local est appliqué à chaque fois qu'une image clef I^i est ajoutée à la reconstruction 3D. Les paramètres des points 3D et ceux des caméras à la fin de la séquence sont raffinés par la minimisation de la fonction de coût :

$$g^i(\mathcal{C}^i, \mathcal{P}^i) = \sum_{C^k \in \{C^{i-N+1} \ \ldots\ C^i\}} \sum_{\mathbf{P}_j \in \mathcal{P}^i} \left\|\pi\left(\mathtt{R}_j^k \left(\mathtt{R}^{k\top}[\mathtt{I}_3 \mid -\mathbf{t}^k]\bar{\mathbf{P}}_j - T_j \mathbf{s}_j^k\right)\right)\right\|^2$$

où $\mathcal{C}^i$ et $\mathcal{P}^i$ sont respectivement les paramètres des caméras génériques (paramètres extrinsèques aux images clef) et les points 3D choisis à l'étape i (Figure 4.3). Le critère minimisé tient compte des reprojections dans les N (avec $N \geq n$) dernières images clef (typiquement $n = 3$ et $N = 10$).

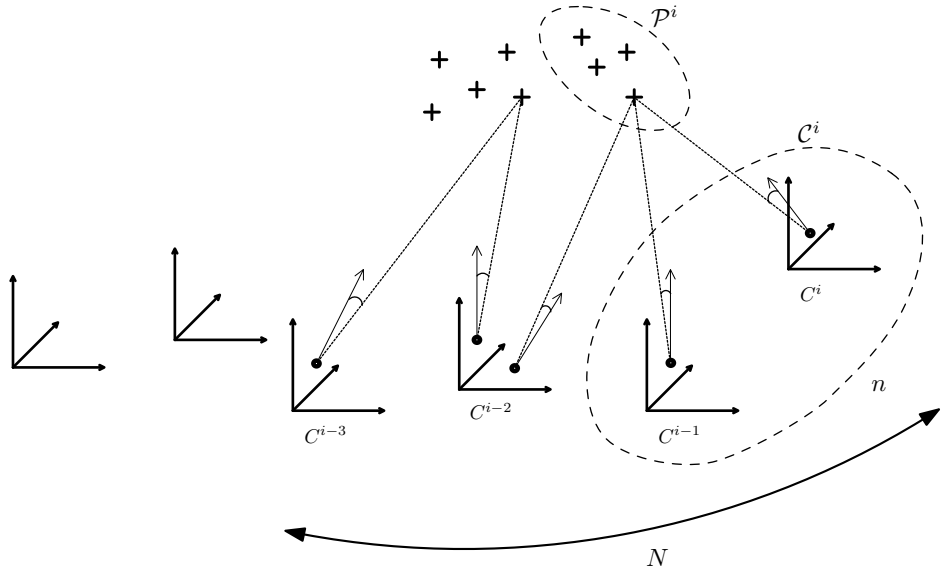
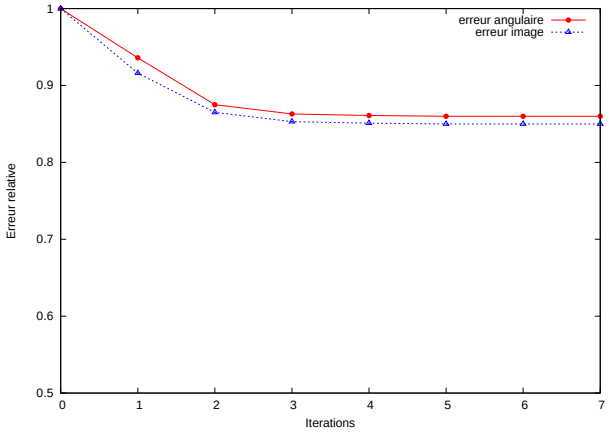


Fig. 4.3 – Ajustement de faisceaux local et angulaire appliqué lorsque la i-ème image clef (de pose C^i) est sélectionnée. Seuls les points $\mathcal{P}^i$ et poses $\mathcal{C}^i$ entourées sont optimisées. La fonction de coût minimisée tient compte des projections des points 3D dans les N dernières images clef.

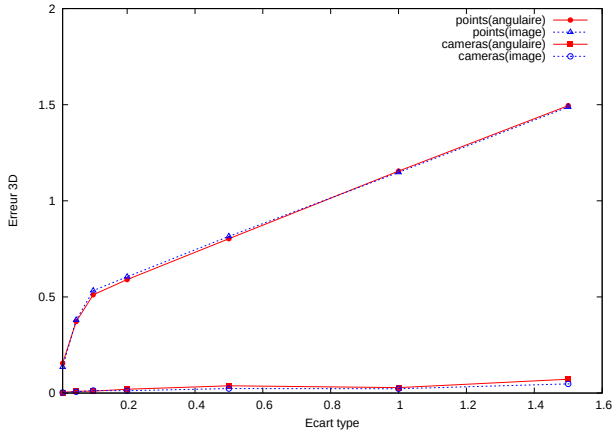
4.3.3 Comparaison Erreur angulaire/Erreur image pour une caméra perspective

Avant tout, il est nécessaire de vérifier que l'ajustement de faisceaux générique angulaire ne dégrade pas les performances de l'ajustement de faisceaux classique, tant au niveau de la précision qu'au niveau de la vitesse de convergence. Les résultats de la comparaison sont présentés sur la Figure 4.4. Le graphique 4.4(a) montre la décroissance moyenne de l'erreur au cours des itérations, dans le cas classique *erreur de reprojection image* et dans le cas *erreur angulaire*. Les premiers tests ont été effectués sur 4 exemples (Tableau 4.4(c)) sur des données réelles non-triviales (2 en intérieur et 2 en extérieur) acquises par une caméra perspective (images de taille 512×384). On voit que dans les deux cas, l'erreur évolue quasiment de la même manière, ce qui

montre que le choix de l'erreur angulaire est correct en ce qui concerne la convergence.
Une seconde expérience a été menée sur des données de synthèse. On part d'une reconstruction exacte à laquelle on ajoute un bruit gaussien dans l'espace image. Ensuite, on raffine globalement de deux façons différentes : en minimisant une erreur image ou une erreur angulaire. Le graphique 4.4(b) montre l'erreur 3D commise sur la position 3D des points et des centres optiques des caméras par rapport à la reconstruction exacte, en fonction de l'écart type sur le bruit. On voit que l'erreur angulaire n'est pas plus sensible au bruit que l'erreur de reprojection dans l'image. Finalement, on remarque que les temps de calculs sont très similaires dans les deux cas.



(a) Progression moyenne de l'erreur $||\epsilon||/||\epsilon_0||$ en fonction des itérations.



(b) Erreur 3D sur la position des points et des caméras en fonction de l'écart type σ du bruit ajouté sur les données.

Séq.	#img	#pts 3D	#pts 2D	Temps img (s)	Temps angl (s)
Ex. 1	30	1.889	7.855	1.9	2.2
Ex. 2	30	1.836	6.948	1.2	1.3
Ex. 3	30	1.061	4.792	1.9	1.9
Ex. 4	30	1.312	6.025	1.8	1.8

(c) Données et temps de calcul.

FIG. 4.4 – Comparaison des ajustements de faisceaux angulaire/image. (a) Convergence. (b) Résistance au bruit. (c) Données et temps de calcul.

4.4 Initialisation générique

Les paragraphes suivants décrivent l'initialisation générique de la reconstruction 3D. Tout d'abord, on présente les coordonnées de Plücker utilisées pour décrire les rayons 3D dans l'espace (section 4.4.1). Ensuite, la notion de contrainte épipolaire généralisée (ou équation de Pless) est introduite dans la section 4.4.2 et les méthodes de résolution adoptées dans différents cas dans la section 4.4.3. Finalement l'initialisation robuste avec trois vues et l'estimation de pose robuste sont respectivement expliquées dans les sections 4.4.4 et 4.4.5.

4.4.1 Les coordonnées de Plücker

Étant donné un ensemble de correspondances de points entre deux images, la pose relative $(\mathtt{R}, \mathbf{t})$ de la deuxième caméra par rapport à la première doit être estimée dans le contexte générique. Pour chaque correspondance 2D (x_0, y_0) et (x_1, y_1) entre deux images 0 et 1, nous connaissons la correspondance de rayons optiques $(\mathbf{s}_0, \mathbf{d}_0)$ et $(\mathbf{s}_1, \mathbf{d}_1)$ dans le repère local de la caméra générique car celle-ci est calibrée. Un rayon $(\mathbf{s}, \mathbf{d})$ est défini par ses coordonnées de Plücker, particulièrement adaptées à ce type de calcul.
Les coordonnées de Plücker d'une droite 3D (ou d'un rayon) sont un couple de vecteurs de dimension 3×1 $(\mathbf{q}, \mathbf{q}')$ respectivement appelés direction et moment. $\mathbf{q}$ donne la direction de la droite et $\mathbf{q}'$ est tel que $\mathbf{q}\cdot\mathbf{q}' = 0$. Chaque point $\mathbf{P} = (x, y, z)$ appartenant à la droite décrite par $(\mathbf{q}, \mathbf{q}')$ satisfait $\mathbf{q}' = \mathbf{q} \wedge \mathbf{P}$. Les coordonnées de Plücker sont définies à un facteur multiplicatif près et une paramétrisation de la droite est donnée par

$$(\mathbf{q}' \wedge \mathbf{q}) + \alpha\mathbf{q}, \ \forall\alpha \in \mathbb{R}. \tag{4.6}$$

Si $||\mathbf{q}|| = 1$, le point $(\mathbf{q}\wedge\mathbf{q}')$ est le point de la droite le plus proche de l'origine du repère et α est la distance signée à ce point.
Pour un rayon optique $(\mathbf{s}, \mathbf{d})$ comme défini précédemment, où $\mathbf{s}$ est l'origine du rayon et $\mathbf{d}$ sa direction dans le repère caméra, les coordonnées de Plücker dans le même repère sont : $\begin{cases} \mathbf{q} = \mathbf{d} \\ \mathbf{q}' = \mathbf{d} \wedge \mathbf{s} \end{cases}$

4.4.2 Contrainte épipolaire généralisée (ou équation de Pless)

Considérons la caméra 0 comme étant à l'origine du repère global. Sa pose s'écrit $(\mathtt{R}_0, \mathbf{t}_0) = (\mathtt{I}_3, \begin{bmatrix}0 & 0 & 0\end{bmatrix}^\top)$. On cherche alors à déterminer $(\mathtt{R}, \mathbf{t})$ qui est la pose de la caméra 1 dans le même repère. Pour chaque correspondance de pixel entre l'image 0 et l'image 1, les rayons optiques $(\mathbf{q}_0, \mathbf{q}'_0)$ et $(\mathbf{q}_1, \mathbf{q}'_1)$ correspondants doivent se couper en un unique point $\mathbf{M}$ de l'espace (voir figure 4.5).

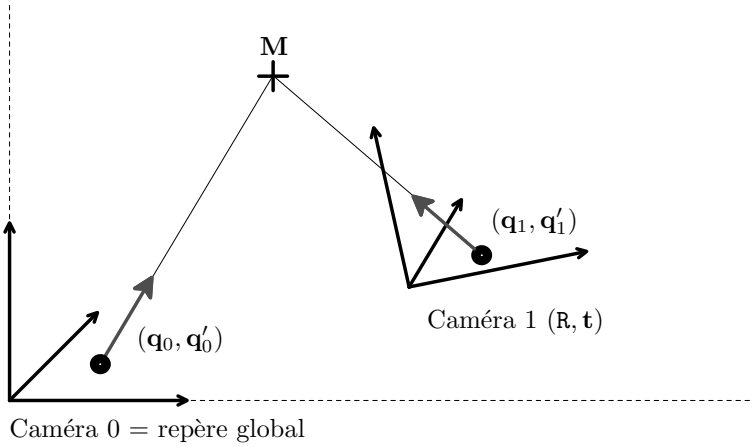


FIG. 4.5 – Pose relative de 2 caméras génériques.

Dans le repère global, les deux rayons sont :

$$\begin{cases} \mathbf{q}_{00} = \mathbf{q}_0 \\ \mathbf{q}'_{00} = \mathbf{q}'_0 \end{cases} \text{ et } \begin{cases} \mathbf{q}_{10} = \mathtt{R}\mathbf{q}_1 \\ \mathbf{q}'_{10} = -[\mathbf{t}]_\times \mathtt{R}\mathbf{q}_1 + \mathtt{R}\mathbf{q}'_1 \end{cases} \tag{4.7}$$

Les deux rayons s'intersectent si et seulement si $\mathbf{q}_{10} \cdot \mathbf{q}'_{00} + \mathbf{q}'_{10} \cdot \mathbf{q}_{00} = 0$, ce qui conduit à la contrainte épipolaire généralisée ou équation de PLESS [71] :

$$\mathbf{q}_0'^{\top} \mathtt{R}\mathbf{q}_1 - \mathbf{q}_0^\top [\mathbf{t}]_\times \mathtt{R}\mathbf{q}_1 + \mathbf{q}_0^\top \mathtt{R}\mathbf{q}'_1 = 0. \tag{4.8}$$

La matrice essentielle généralisée $\mathtt{G}$ est définie comme suit [77] :

$$\mathtt{G} = \begin{pmatrix} -[\mathbf{t}]_\times \mathtt{R} & \mathtt{R} \\ \mathtt{R} & \mathtt{0}_{3\times 3} \end{pmatrix} \tag{4.9}$$

$\mathtt{G}$ est de taille 6×6 et vérifie la contrainte

$$\begin{pmatrix} \mathbf{q}_0 \\ \mathbf{q}'_0 \end{pmatrix}^\top \mathtt{G} \begin{pmatrix} \mathbf{q}_1 \\ \mathbf{q}'_1 \end{pmatrix} = 0 \tag{4.10}$$

si et seulement si les rayons $(\mathbf{q}_0, \mathbf{q}'_0)$ et $(\mathbf{q}_1, \mathbf{q}'_1)$ s'intersectent dans l'espace. $\mathtt{G}$ contient 9 coefficients nuls et 9 coefficients de $\mathtt{R}$ en double. Il y a donc 18 coefficients utiles.

Nous avons identifié deux cas où cette équation a une infinité de solutions. Le premier cas est évident : il y a une infinité de solutions si la caméra est centrale (la reconstruction 3D est calculée à un facteur d'échelle près). On note que l'équation 4.8 est la contrainte épipolaire classique définie par la matrice essentielle $\mathtt{E} = [\mathbf{t}]_\times \mathtt{R}$ si le centre optique de la caméra est à l'origine du repère caméra.

Le second cas est moins évident mais il se produit fréquemment en pratique. Dans nos expériences, nous supposons que nous avons seulement des mises en correspondance "simples" : tous les rayons $(\mathbf{s}^i, \mathbf{d}^i)$ d'un point 3D donné passent par un même centre de projection de la caméra (dans le repère local à la caméra générique). Autrement dit, on a $\mathbf{q}'_0 = \mathbf{q}_0 \wedge \mathbf{c}^0$ et $\mathbf{q}'_1 = \mathbf{q}_1 \wedge \mathbf{c}^1$ avec $\mathbf{c}^0 = \mathbf{c}^1$. Pour un système composé de plusieurs caméras centrales (par exemple la paire stéréo rigide), cela signifie que les points 2D sont seulement mis en correspondance avec d'autres points de la même sous image. Ceci est fréquemment le cas en pratique pour deux raisons : des régions d'intérêt peu étendues pour une mise en correspondance fiable, et des intersections de champs de vue vides pour deux caméras composantes du système multi-caméras. Si le mouvement de la caméra générique est une translation pure ($\mathtt{R} = \mathtt{I}_3$), l'équation 4.8 devient $\mathbf{q}_0^\top [\mathbf{t}]_\times \mathbf{q}_1 = {\mathbf{q}'_0}^\top \mathbf{q}_1 + \mathbf{q}_0^\top \mathbf{q}'_1 = 0$ avec pour inconnue $\mathbf{t}$. Dans ce contexte, l'échelle de $\mathbf{t}$ ne peut être estimée. On supposera dans ces travaux que le mouvement de caméra n'est pas une translation pure au début de la séquence d'images.

4.4.3 Résolution de l'équation de Pless

On re-écrit l'équation 4.8 sous la forme

$$ {\mathbf{q}'_0}^\top \tilde{\mathtt{R}} \mathbf{q}_1 - \mathbf{q}_0^\top \tilde{\mathtt{E}} \mathbf{q}_1 + \mathbf{q}_0^\top \tilde{\mathtt{R}} \mathbf{q}'_1 = 0 \tag{4.11} $$

avec les deux matrices $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}})$ de dimension 3×3 comme nouvelles inconnues. On peut ranger les coefficients de $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}})$ dans un vecteur $\mathbf{x}$ à 18 dimensions et on voit que chaque valeur du 4-uplet $(\mathbf{q}_0, \mathbf{q}'_0, \mathbf{q}_1, \mathbf{q}'_1)$ donne une équation linéaire $\mathbf{a}^\top \mathbf{x} = 0$. Si l'on dispose de 17 couples de points appariés, alors on a 17 valeurs de ce 4-uplet et donc 17 équations $\mathbf{a}_k^\top \mathbf{x} = 0$, $k = 1 \ldots 17$.
Cela suffit pour déterminer $\mathbf{x}$ à un facteur d'échelle près [77]. On construit la matrice $\mathtt{A}_{17}$ contenant les 17 mises en correspondance telles que le système à

résoudre s'écrive

$$\|\mathtt{A}_{17}\mathbf{x}\| = 0 \tag{4.12}$$

avec $\mathtt{A}_{17}^{\top} = [\mathbf{a}_1^{\top}|\mathbf{a}_2^{\top}|\cdots\mathbf{a}_{17}^{\top}]$. La résolution dépend de la dimension du noyau de $\mathtt{A}_{17}$, qui dépend lui-même du type de caméra utilisée. Nous déterminons $Ker(\mathtt{A}_{17})$ et sa dimension par une décomposition en valeurs singulières de $\mathtt{A}_{17}$. On distingue plusieurs cas dans la thèse : (1) caméra centrale à un seul centre optique (2) caméra axiale où les centres sont alignés (ce qui inclut la paire stéréo) et (3) les caméras non axiales.

Il n'est pas surprenant que la dimension du noyau du système linéaire à résoudre soit plus grand que 1. En effet, l'équation linéaire 4.11 a plus d'inconnues (18 inconnues) que l'équation non-linéaire 4.8 (6 inconnues). Des dimensions possibles sont données dans le Tableau 4.1 (sous notre hypothèse de mise en correspondance "simple") et sont justifiées dans les paragraphes qui suivent. Les travaux précédents [71, 77] ignorent ces dimensions, bien qu'une méthode (éventuellement linéaire) devrait en tenir compte.

Caméra	Centrale	Axiale	Non axiale
$dim(Ker(\mathtt{A}_{17}))$	10	4	2

TAB. 4.1 – $dim(Ker(\mathtt{A}_{17}))$ dépend du type de caméra.

Caméra centrale

Pour les caméras centrales (par exemple perspectives ou catadioptriques à centre unique), tous les rayons optiques convergent vers le centre optique $\mathbf{c}$. Puisque $\mathbf{q}_i' = \mathbf{q}_i \wedge \mathbf{c} = [-\mathbf{c}]_{\times}\mathbf{q}_i$, l'équation 4.11 devient $\mathbf{q}_0{}^{\top}([\mathbf{c}]_{\times}\tilde{\mathtt{R}} - \tilde{\mathtt{E}} - \tilde{\mathtt{R}}[\mathbf{c}]_{\times})\mathbf{q}_1 = 0$. On note que $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}}) = (\tilde{\mathtt{R}}, [\mathbf{c}]_{\times}\tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}]_{\times})$ est une solution de l'équation 4.11 pour toute matrice $\tilde{\mathtt{R}}$. De telles solutions sont "exactes" : l'équation 4.11 est exactement égale à 0 (à la précision de l'ordinateur près) quelque soit $(\mathbf{q}_0, \mathbf{q}_1)$. Notre "vraie" solution est $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}}) = (\mathtt{0}, [\mathbf{t}]_{\times}\mathtt{R})$ si $\mathbf{c} = (0,0,0)^{\top}$, et l'égalité n'est pas exacte à cause du bruit dans les images. Donc, la dimension de $Ker(\mathtt{A}_{17})$ est au moins $9+1$. Les expériences confirment que cette dimension est 10 (au bruit près).

Soit $\mathtt{A}_{17} = \mathtt{UDV}^{\top}$ la décomposition SVD de $\mathtt{A}_{17}$. Dans le cas où $\mathbf{c} = [0\ 0\ 0]^{\top}$, on remarque que le vecteur $\mathbf{y}$ de la solution non exacte (9^{e} colonne de $\mathtt{V}$, 10^{e} colonne en partant de la fin) est orthogonal à chacun des vecteurs des solutions exactes (9 dernières colonnes de $\mathtt{V}$). Comme le vecteur $\mathbf{y}$ est dans le noyau de dimension 10 de $\mathtt{A}_{17}$, $\mathbf{y}$ est combinaison linéaire des 10 dernières

colonnes de $\mathtt{V}$. On en déduit que $\mathbf{y}$ est égal (à un coefficient multiplicatif près) à la 9$^{\text{e}}$ colonne de $\mathtt{V}$. On note également que $\mathbf{y}$ est la solution de l'équation épipolaire habituelle $\mathbf{q}_0{}^\top[\mathbf{t}]_\times \mathtt{R}\mathbf{q}_1 = 0$. Pour résoudre, on extrait d'abord les coefficients de la matrice $\tilde{\mathtt{E}}$ qui sont contenus dans la 9$^{\text{e}}$ colonne de $\mathtt{V}$. Ensuite, on en déduit les 4 solutions pour la pose relative $(\mathtt{R}, \mathbf{t})$ entre les deux points de vue, comme expliqué dans la section 1.3.1. Parmi les 4 solutions, on choisit celle qui permet de trianguler les 17 points devant les 2 caméras. Si un point 3D a pour coordonnées $\mathbf{P}$ dans le repère de la caméra, et si le rayon optique qui lui correspond est $(\mathbf{s}, \mathbf{d})$, alors on considère que ce point est devant la caméra si et seulement si $(\mathbf{P} - \mathbf{s}) \cdot \mathbf{d} > 0$.

Caméra axiale

Ce cas inclut la paire stéréo classique composée de deux caméras perspectives rigidement liées. Soient $\mathbf{c}_a$ et $\mathbf{c}_b$ deux points sur l'axe de la caméra (contenant tous les centres). L'Annexe 2 montre que des solutions exactes sont définies par

$$\begin{aligned}&\tilde{\mathtt{E}} = [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_a]_\times \text{ et}\\ &\tilde{\mathtt{R}} \in Vect\{\mathtt{I}_{3\times 3}, [\mathbf{c}_a - \mathbf{c}_b]_\times, (\mathbf{c}_a - \mathbf{c}_b)(\mathbf{c}_a - \mathbf{c}_b)^\top\}\end{aligned}$$

avec nos hypothèses d'appariements "simples" (section 4.4.2). La "vraie" solution n'est pas exacte à cause du bruit dans les images, et donc la dimension de $Ker(\mathtt{A}_{17})$ est au moins $3+1$. Les expériences confirment que cette dimension est 4 (au bruit près).

Nous construisons une base de 3 solutions exactes $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ et une solution non exacte $\mathbf{y}$ avec les vecteurs singuliers correspondants aux 4 plus petites valeurs singulières de $\mathtt{A}_{17}$. Les valeurs singulières associées à $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ sont 0 à la précision de l'ordinateur près et celle de $\mathbf{y}$ est 0 au bruit image près. Nous recherchons la solution réelle $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}})$ par combinaison linéaire de $\mathbf{y}$, $\mathbf{x}_1$, $\mathbf{x}_2$ et $\mathbf{x}_3$ telle que la matrice résultante $\tilde{\mathtt{R}}$ vérifie $\tilde{\mathtt{R}}^\top\tilde{\mathtt{R}} = \lambda\mathtt{I}_{3\times 3}$ ou $\tilde{\mathtt{E}}$ est une matrice essentielle. Soit $\mathbf{l}$ un vecteur tel que $\mathbf{l}^\top = [\lambda_1\ \lambda_2\ \lambda_3]^\top$. On note alors $\tilde{\mathtt{R}}(\mathbf{l})$ et $\tilde{\mathtt{E}}(\mathbf{l})$ les matrices $\tilde{\mathtt{R}}$ et $\tilde{\mathtt{E}}$ extraites de la solution $\mathbf{y} - [\mathbf{x}_1|\mathbf{x}_2|\mathbf{x}_3]\mathbf{l}$. Avec ces notations, on a $\tilde{\mathtt{R}}(\mathbf{l}) = \mathtt{R}_0 - \sum_{i=1}^3 \lambda_i \mathtt{R}_i$ et $\tilde{\mathtt{E}}(\mathbf{l}) = \mathtt{E}_0 - \sum_{i=1}^3 \lambda_i \mathtt{E}_i$ avec $(\mathtt{R}_i, \mathtt{E}_i)$ extraites de $\mathbf{x}_i$ et $(\mathtt{R}_0, \mathtt{E}_0)$ extraites de $\mathbf{y}$.

Une fois la base $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ calculée, nous calculons les coordonnées de la solution par minimisation non-linéaire de la fonction $(\lambda, \mathbf{l}) \rightarrow \|\lambda\mathtt{I}_{3\times 3} - \mathtt{R}(\mathbf{l})^\top\mathtt{R}(\mathbf{l})\|^2$ pour obtenir $\mathbf{l}$ et donc $\tilde{\mathtt{E}}$. Une décomposition SVD est appliquée sur $\tilde{\mathtt{E}}$ et on obtient 4 solutions [30] pour $([\mathbf{t}]_\times, \mathtt{R})$. La solution avec une contrainte épipolaire minimale $\|\mathtt{A}_{17}\mathbf{x}\|$ est choisie. Enfin, on raffine le facteur

d'échelle k en minimisant $k \rightarrow \sum_i ({\mathbf{q}'_{0i}}^\top \mathtt{R}\mathbf{q}_{1i} - \mathbf{q}_{0i}^\top k[\mathbf{t}]_\times \mathtt{R}\mathbf{q}_{1i} + \mathbf{q}_{0i}^\top \mathtt{R}\mathbf{q}'_{1i})^2$ et on applique $\mathbf{t} \leftarrow k\mathbf{t}$.

Caméra non axiale

Pour les caméras non axiales (par exemple un système de trois caméras perspectives rigidement liées dont les centres ne sont pas alignés), le problème est encore différent. L'annexe montre que des solutions "exactes" sont $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}}) \in Vect\{(\mathtt{I}_{3\times3}, \mathtt{0}_{3\times3})\}$ avec notre hypothèse d'appariements "simples" (section 4.4.2). La "vraie" solution n'est pas exacte à cause du bruit dans les images, et donc la dimension de $Ker(\mathtt{A}_{17})$ est au moins $1+1$. Les expériences confirment que cette dimension est 2 (au bruit près). Nous n'avons pas expérimenté ce cas sur un dispositif réel.

4.4.4 Initialisation robuste pour trois vues

La première étape de la méthode est la reconstruction 3D du premier triplet d'images clef $\{0, 1, 2\}$ avec une méthode de type RANSAC. On effectue plusieurs tirages aléatoires de 17 couples de points appariés. Pour chaque tirage, la pose relative entre les vues 0 et 2 est calculée avec la méthode qui précède et les points appariés sont reconstruits par triangulation. La pose de la caméra 1 est estimée à partir des correspondances 3D/2D par raffinement itératif en minimisant l'erreur angulaire définie dans la section 4.3.2 (plus de détails sur le calcul de pose dans la section 4.4.5). La même erreur est minimisée pour trianguler les points. Finalement, la solution (parmi celles obtenues avec tous les tirages) qui donne le maximum de points consistants avec la géométrie des points de vues 0, 1, et 2 est retenue. Le j-ème point 3D est consistant avec la i-ème vue si l'erreur angulaire $||\epsilon_j^i||$ est inférieure à un seuil ϵ ($\epsilon = 0.01\ rad$ dans nos expériences).

4.4.5 Estimation de pose robuste

Le calcul de pose générique est utile pour les deux étapes de notre méthode (initialisation et processus incrémental). On suppose que la i-ème pose $C^i = (\mathtt{R}^i, \mathbf{t}^i)$ de la caméra est proche de la $i-1$-ème pose $C^{i-1} = (\mathtt{R}^{i-1}, \mathbf{t}^{i-1})$. C^i est estimée par optimisation non-linéaire initialisée à C^{i-1} avec un nombre réduit n (pas nécessairement minimal) de mises en correspondance 3D/2D (par exemple n=5), en conjonction avec la méthode robuste RANSAC. Une mise en correspondance 3D/2D est un point suivi sur 3 images et reconstruit

sur 2 d'entre elles. Pour chaque tirage aléatoire de n correspondances 3D/2D, la pose est estimée en minimisant une erreur angulaire (section 4.3.2) par LEVENBERG-MARQUARDT, et nous comptons ensuite le nombre de points 3D consistants avec cette pose. La pose avec le maximum de points consistants est retenue, et un raffinement de cette pose est effectué avec tous les points qui ont été déclarés consistants à l'étape précédente.

4.5 Résultats

L'algorithme de reconstruction 3D générique a été testé sur des données réelles avec trois différents types de caméras : une caméra perspective, une caméra catadioptrique et une paire stéréo. Des exemples d'images sont visibles sur la Figure 4.6 et les caractéristiques des séquences vidéo sont données dans le Tableau 4.2. Les performances en temps de calcul sont reportées dans le Tableau 4.3. A la suite de ces expériences, la trajectoire obtenue avec notre méthode générique est comparée à la vérité terrain GPS et à une trajectoire obtenue par ajustement de faisceaux global et spécifique. Comme précédemment, une transformation rigide (rotation, translation et mise à l'échelle) est appliquée à la trajectoire afin de la recaler au mieux avec les données de référence et calculer l'erreur résultante.

4.5.1 Comparaison avec la vérité terrain (GPS différentiel)

Les premiers résultats correspondent à la séquence vidéo présentée dans la section 3.7.2, avec la caméra perspective embarquée sur le *Cycab*. La trajectoire calculée avec notre méthode incrémentale générique est comparée avec les données provenant du capteur GPS différentiel, et la Figure 4.7 montre les deux trajectoires recalées. On mesure l'erreur moyenne de positionnement : l'erreur 3D est de 0.48 m et l'erreur dans le plan horizontal est de 0.47 m. Les résultats sont comparables (très légèrement moins précis) à ceux obtenus avec la méthode spécifique aux caméras perspectives.

4.5.2 Comparaison avec un ajustement de faisceaux global et spécifique

Dans les exemples qui suivent, la vérité terrain n'est pas donnée. On compare donc nos résultats avec ceux de la meilleure méthode connue : un

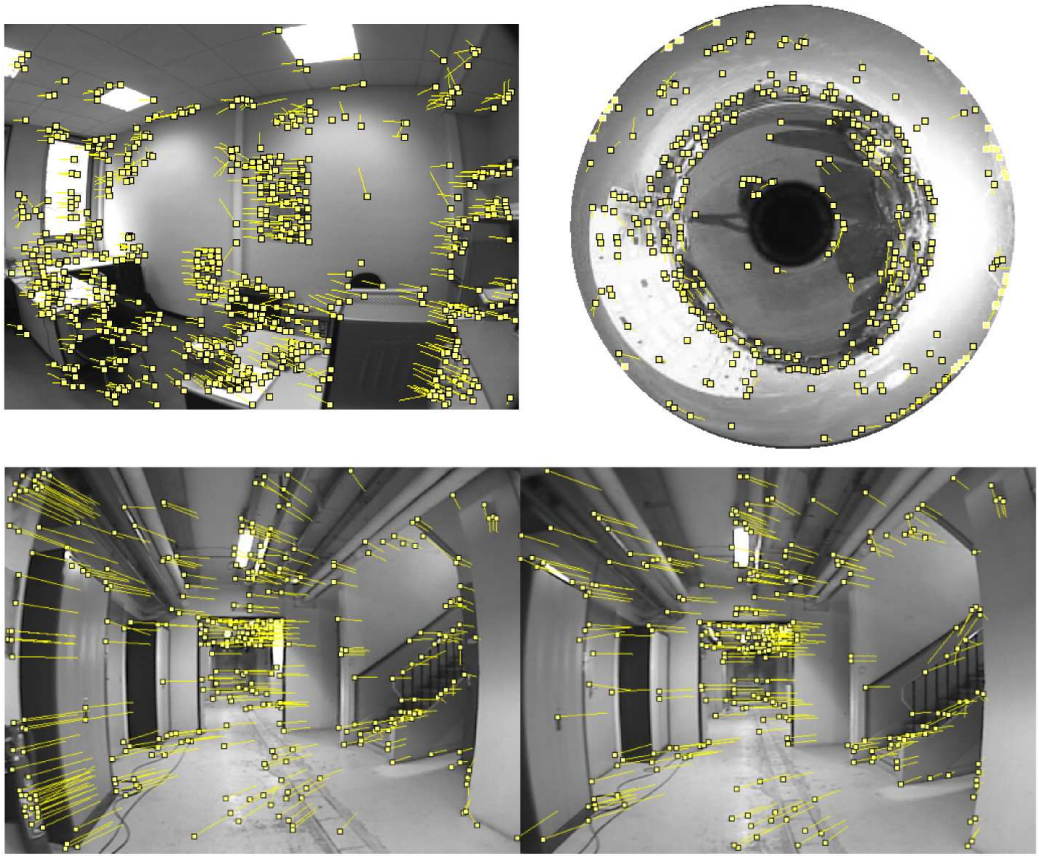


FIG. 4.6 – Suivi de points pour une image de caméra générique dans trois cas : caméra perspective (en haut a gauche), caméra catadioptrique (en haut a droite) et paire stéréo (en bas). La méthode de mise en correspondance est celle décrite au chapitre 3 quel que soit le type de caméra utilisé.

ajustement de faisceaux global et spécifique (tous les paramètres de la séquence complète sont raffinés en minimisant la somme des carrés des erreurs de reprojection). Les caractéristiques et résultats sur les séquences sont donnés dans le Tableau 4.2.

La séquence 2 est prise dans un environnement intérieur avec une caméra perspective tenue à la main. Le résultat obtenu est comparable à la reconstruction optimisée globalement : l'erreur 3D moyenne est inférieure à 6.5 cm pour une trajectoire d'environ 15 m. L'erreur relative est 0.45%.

La séquence 3 est prise dans un environnement extérieur avec une caméra catadioptrique tenue à la main (le miroir "0-360" avec la caméra Sony HDR-

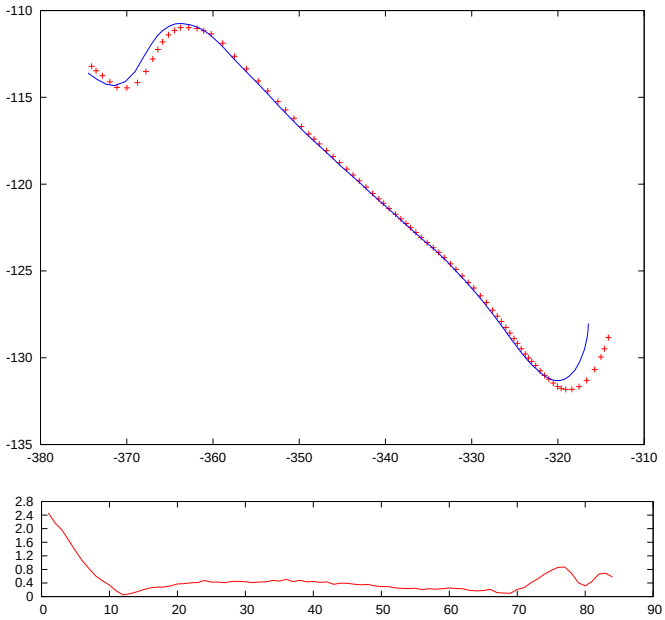


FIG. 4.7 – Comparaison avec la vérité terrain GPS. haut : recalage avec la trajectoire estimée (méthode générique locale). La ligne continue donne la trajectoire GPS et les points représentent les positions estimées aux images clef. bas : l'erreur 3D le long de la trajectoire.

HC1E en utilisant le format DV, visibles sur la Figure 4.8). La partie utile de l'image rectifiée est contenue dans un disque dont le diamètre est de 458 *pixels*. Le résultat est également précis : l'erreur 3D moyenne est inférieure à 9.2 *cm* pour une trajectoire (d'environ) 40 *m*. L'erreur relative est 0.23%.

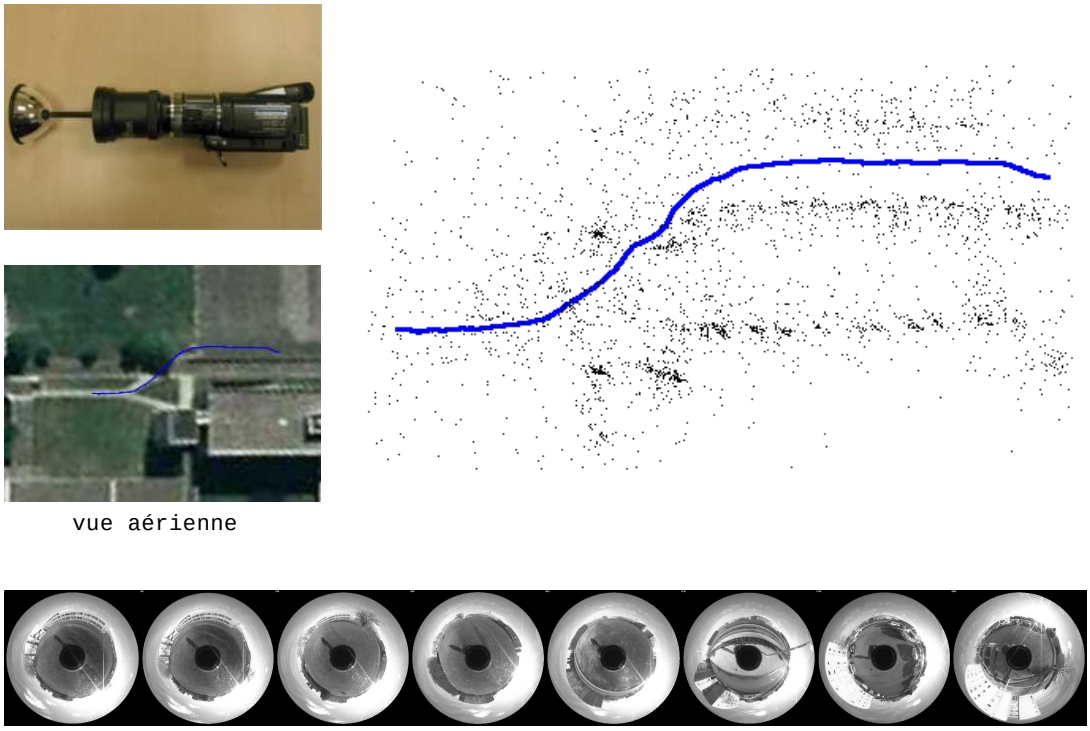
La séquence 4 est prise avec une paire stéréo (la baseline est de 40 *cm*) dans un couloir (Figure 4.8). L'image est composée de deux sous-images de dimensions 640×480. La trajectoire (longue de 20 *m*) est comparée avec les résultats obtenus avec un ajustement de faisceaux global appliqué sur la caméra de gauche/droite. L'erreur 3D moyenne est 2.7/8.4 *cm* comparée à la caméra de gauche/droite, et l'erreur relative est de 0.13%/0.42%.

Séquence	Caméra	#img	#img clef	#Pts 3D	#Pts 2D	Long. traj. (m)
Séq 1	perspective	445	84	4.477	19.502	70
Séq 2	perspective	511	48	3.162	11.966	15
Séq 3	catadioptrique	1.493	132	4.752	18.198	40
Séq 4	paire stéréo	303	28	3.642	14.189	20

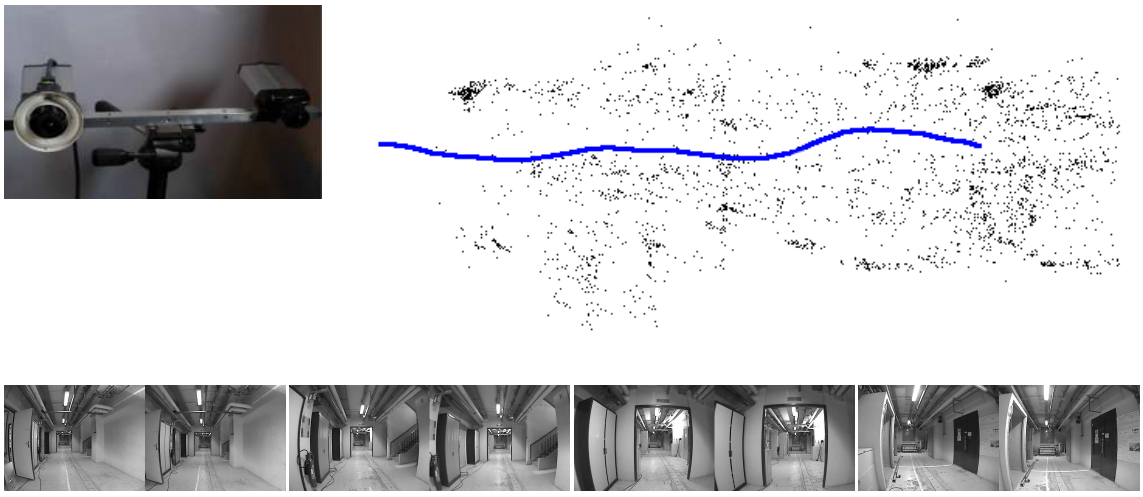
TAB. 4.2 – Caractéristiques des séquences vidéo.

Caméra	Image	Détect.+App.	images	images clef	fréquence
Perspective	512×384	0.10	0.14	0.37	**6.3 fps**
Catadioptrique	464×464	0.12	0.15	0.37	**5.9 fps**
Paire stéreo	1280×480	0.18	0.25	0.91	**3.3 fps**

TAB. 4.3 – Temps de calcul en secondes pour les trois caméras. Les temps de Détect.+App. sont inclus dans les temps images/images clef .



(a) Séquence 3 : caméra catadioptrique (extérieur).



(b) Séquence 4 : caméra stéréo (couloir).

FIG. 4.8 – Caméras, images des séquences vidéo et reconstructions 3D vues de dessus (trajectoires en bleu et points 3D en noir).

Conclusion et perspectives

Notre objectif dans cette thèse était de développer une méthode de reconstruction 3D et localisation pour une ou plusieurs caméras en mouvement. Les contraintes étaient fortes : une rapidité de calcul permettant un fonctionnement en temps-réel et une bonne précision pour traiter des mouvements de caméras de plusieurs centaines de mètres sans entraîner de forte dérive. Dans un premier temps, pour parvenir à ce résultat, une méthode de reconstruction incrémentale a été développée pour une caméra perspective. Avec cette méthode, la géométrie 3D est reconstruite séquentiellement à partir d'images clef et optimisée localement par ajustement de faisceaux. L'idée d'employer l'ajustement de faisceaux était tout à fait logique puisque l'on était à la recherche de précision. Elle l'était beaucoup moins dans le but de faire des calculs rapides. Sous sa forme locale, l'ajustement de faisceaux optimise un nombre de paramètres limité, ce qui fait diminuer la complexité de calcul. Les résultats ont montré que l'on peut grandement accélérer les calculs par rapport à l'ajustement de faisceaux global tout en ayant une perte de précision minime. On peut donc en conclure que l'ajustement de faisceaux peut faire partie intégrante d'un système temps-réel.

Les nombreuses applications possibles évoquées ouvrent de grandes perspectives. Dans le domaine de l'automobile, l'odométrie visuelle peut complémenter les autres solutions de localisation (GPS, odométrie des roues, centrale inertielle, etc) par fusion des données pour tirer le meilleur de chaque capteur suivant le contexte. De plus, le calcul par vision de la structure 3D de la scène peut s'avérer utile, comme on l'a vu pour l'application de créneau automatique ou encore pour la détection d'obstacles. Une perspective proche sur le thème des robots mobiles intelligents, est le convoi par vision. Au problème de localisation s'ajoutent alors les problèmes de transmission des données sans fil et de synchronisation des véhicules. On peut également penser à appliquer ces techniques pour l'aide au déplacement des personnes malvoyantes en développant des systèmes de guidage portables équipés de caméras.

Pour toutes ces applications, de grandes améliorations de la méthode sont réalisables :

- Le problème de la dérive reste présent même si il a été limité. Il serait intéressant d'étudier ses causes et les moyens de l'annuler mais également comment tenir compte des boucles et des problèmes de relocalisation. Pour cela, il est envisageable de combiner la méthode avec des approches de type SLAM.
- Pour les systèmes embarqués (automobile, etc), il semble nécessaire de prévoir une fonction d'autocalibrage. Tout d'abord, on a vu qu'un bon calibrage est important pour la qualité des résultats, mais également parce qu'il est susceptible d'évoluer au cours du temps.
- Des simplifications peuvent être effectuées au niveau de la modélisation du mouvement de la caméra. La méthode prévoit une liberté totale à 6 degrés de liberté. L'idée serait de tenir compte de la cinématique des véhicules et de la non-holonomie des mouvements impliqués.
- En ce qui concerne les temps de calcul, des accélérations sont possibles à partir de la seule amélioration du code source existant. Pour une implémentation "en ligne" il serait intéressant de diminuer les temps de calcul des images clef voire même d'éliminer cette notion d'image clef afin que les temps de calculs soient équitablement répartis entre les images. On veillerait à ce que le temps de calcul pour chaque image reste inférieur au temps disponible.
- A ce jour, nous nous sommes uniquement intéressés à la reconstruction d'un nuage de points. Cependant, pour certaines applications, il serait judicieux d'introduire des primitives plus complexes et mieux interprétables par un utilisateur. Par exemple un plan pour délimiter la façade d'un immeuble ou un cube pour un obstacle sur la route, etc. L'idée de gérer les motifs répétitifs très présents dans les environnements façonnés par l'homme (fenêtres des immeubles, pavages, etc) peut également être explorée.

Dans un deuxième temps, on a étudié la généralisation de la méthode aux caméras génériques. Le principe général de la méthode est resté le même mais des points précis ont été revus et généralisés. Les résultats ont montré que les performances étaient équivalentes à la méthode spécifique d'origine. Outre la facilité d'utilisation et de changement de type de caméra, la modélisation générique était intéressante car elle permettait la modélisation d'un ensemble rigide de caméras. Les applications sont pertinentes en automobile où l'utilisation de plusieurs caméras peut apporter de la robustesse avec des

champs de vues plus importants. Pour cette méthode générique, une amélioration est à prévoir au niveau de la méthode de mise en correspondance afin de la rendre plus efficace. Des expérimentations complémentaires sur données réelles sont également à réaliser. En effet, les cas suivants n'ont pas été expérimentés à ce jour :

- Les caméras à plus de deux centres non alignés.
- Les multi-caméras à champs de vue disjoints, ainsi que leur calibrage.

Annexes

Annexe 1 : calcul des dérivées analytiques et résolution du système creux

Dans cette annexe, on détaille les calculs des dérivées analytiques intervenant dans l'ajustement de faisceaux. Il s'agit des dérivées de la fonction d'erreur minimisée par rapport aux paramètres optimisés (coordonnées des points 3D et paramètres extrinsèques des caméras). On explique techniquement la construction et la résolution du système creux impliqué.

Cas classique : erreur de reprojection image

Rappelons que la projection d'un point 3D $\bar{\mathbf{P}} = [X\ Y\ Z\ T]^\top$ en un point 2D $\mathbf{p} = [x_p\ y_p]^\top$ (de coordonnées homogènes $\bar{\mathbf{p}} = [X_p\ Y_p\ Z_p]^\top$) par la caméra C dont la pose est $(\mathtt{R}, \mathbf{t})$ s'écrit :

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix} = \pi\left(\mathtt{C}\bar{\mathbf{P}}\right) = \pi\left(\mathtt{K}\mathtt{R}^\top\left[\mathtt{I}_3\right| - \mathbf{t}\right]\bar{\mathbf{P}}\right)$$

L'erreur de reprojection 2D pour le point détecté $\mathbf{p} = [x_0\ y_0]^\top$ est :

$$\boldsymbol{\epsilon} = \begin{pmatrix} x_p - x_0 \\ y_p - y_0 \end{pmatrix} = \pi\left(\mathtt{K}\mathtt{R}^\top\left[\mathtt{I}_3\right| - \mathbf{t}\right]\bar{\mathbf{P}}\right) - \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$$

qui peut être représentée par la fonction $f : \mathbb{R}^{10} \rightarrow \mathbb{R}^2$ telle que $\epsilon = f(X, Y, Z, T, t_x, t_y, t_z, \alpha, \beta, \gamma)$ avec α, β et γ les angles d'Euler correspondant à la matrice de rotation $\mathtt{R}$ et t_x, t_y, t_z les composantes de $\mathbf{t}$. On recherche la matrice Jacobienne de f qui est :

$$\mathtt{j} = \underbrace{\left(\frac{\partial f}{\partial \bar{\mathbf{P}}} \mid \frac{\partial f}{\partial \mathbf{t}} \mid \frac{\partial f}{\partial \mathtt{R}}\right)}_{(2\times4)|(2\times3)|(2\times3)} = \begin{pmatrix} \frac{\partial f_x}{\partial X} & \frac{\partial f_x}{\partial Y} & \frac{\partial f_x}{\partial Z} & \frac{\partial f_x}{\partial T} & \frac{\partial f_x}{\partial t_x} & \frac{\partial f_x}{\partial t_y} & \frac{\partial f_x}{\partial t_z} & \frac{\partial f_x}{\partial \alpha} & \frac{\partial f_x}{\partial \beta} & \frac{\partial f_x}{\partial \gamma} \\ \\ \frac{\partial f_y}{\partial X} & \frac{\partial f_y}{\partial Y} & \frac{\partial f_y}{\partial Z} & \frac{\partial f_y}{\partial T} & \frac{\partial f_y}{\partial t_x} & \frac{\partial f_y}{\partial t_y} & \frac{\partial f_y}{\partial t_z} & \frac{\partial f_y}{\partial \alpha} & \frac{\partial f_y}{\partial \beta} & \frac{\partial f_y}{\partial \gamma} \end{pmatrix}$$

Puisque f est une fonction composée, on a

$$d(f) = d(\pi) \ \times \ d\left(\mathtt{KR}^\top \left[\mathtt{I}_3 | -\mathbf{t}\right] \bar{\mathbf{P}}\right)$$

et si $\pi\left([X_p \ Y_p \ Z_p]^\top\right) = [X_p/Z_p \ Y_p/Z_p]^\top$, alors $d(\pi) = \begin{pmatrix} 1/Z_p & 0 & -X_p/Z_p^2 \\ 0 & 1/Z_p & -Y_p/Z_p^2 \end{pmatrix}$.

Il reste maintenant à calculer les dérivées de $\mathtt{KR}^\top \left[\mathtt{I}_3 | -\mathbf{t}\right] \bar{\mathbf{P}}$ par rapport à $X, Y, Z, T, \alpha, \beta$ et γ.

Dérivées par rapport aux paramètres des points 3D

Elles s'obtiennent immédiatement en dérivant par rapport à $\bar{\mathbf{P}}$:

$$\begin{aligned} \frac{\partial f}{\partial \bar{\mathbf{P}}}(\bar{\mathbf{P}}, \mathbf{t}, \mathtt{R}) &= d(\pi) \ \times \ \frac{\partial}{\partial \bar{\mathbf{P}}} \left(\mathtt{KR}^\top \left[\mathtt{I}_3 | -\mathbf{t}\right] \bar{\mathbf{P}}\right) \\ &= \underbrace{d(\pi) \ \times \ \mathtt{KR}^\top \left[\mathtt{I}_3 | -\mathbf{t}\right]}_{(2\times3)(3\times3)(3\times3)(3\times4)=(2\times4)} \end{aligned}$$

Dérivées par rapport aux paramètres des caméras

1. Dérivées par rapport aux paramètres de translation :

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{t}}(\bar{\mathbf{P}}, \mathbf{t}, \mathtt{R}) &= d(\pi) \ \times \ \frac{\partial}{\partial \mathbf{t}} \left(\mathtt{KR}^\top \left[\mathtt{I}_3 | -\mathbf{t}\right] \bar{\mathbf{P}}\right) \\ &= d(\pi) \ \times \ \mathtt{KR}^\top \ \times \ \frac{\partial}{\partial \mathbf{t}} \left(\left[\mathtt{0}_{3\times3} | -\mathbf{t}\right] \bar{\mathbf{P}}\right) \\ &= -T \times \underbrace{d(\pi) \ \times \ \mathtt{KR}^\top}_{(2\times3)(3\times3)(3\times3)=(2\times3)} \end{aligned}$$

2. Dérivées par rapport aux paramètres de rotation :
   Pour paramétriser la rotation, on utilise un repère relatif au voisinage de la matrice de rotation $\mathtt{R}_0$ courante, telle que $\mathtt{R}_0(\boldsymbol{\omega}) = \mathtt{R}(\boldsymbol{\omega})\mathtt{R}_0$ où $\boldsymbol{\omega} = [\omega_1 \ \omega_2 \ \omega_3]^\top$ contient les petits angles d'Euler autour de $\mathtt{R}_0$ et

$$\mathtt{R}(\boldsymbol{\omega}) = \begin{pmatrix} \cos\omega_3 & -\sin\omega_3 & 0 \\ \sin\omega_3 & \cos\omega_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\omega_2 & 0 & \sin\omega_2 \\ 0 & 1 & 0 \\ -\sin\omega_2 & 0 & \cos\omega_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\omega_1 & -\sin\omega_1 \\ 0 & \sin\omega_1 & \cos\omega_1 \end{pmatrix}.$$

Au voisinage de $\mathtt{R}_0$, $\mathtt{R}(\boldsymbol{\omega})$ est proche de $\mathtt{I}_3$ et on peut montrer que

$$\mathtt{R}(\boldsymbol{\omega}) = \mathtt{I}_3 + [\boldsymbol{\omega}]_\times + o(\boldsymbol{\omega}) \approx \begin{pmatrix} 1 & -\omega_3 & \omega_2 \\ \omega_3 & 1 & -\omega_1 \\ -\omega_2 & \omega_1 & 1 \end{pmatrix}.$$

Ainsi, à chaque itération d'ajustement de faisceaux, les dérivées par rapport aux angles d'EULER α, β, et γ dans le repère global sont remplacées par les dérivées par rapport aux petits angles ω_1, ω_2 et ω_3 dans le repère local lié à la matrice de rotation $\mathtt{R}_0$. Le calcul est fait de la manière suivante :

$$\begin{aligned} \frac{\partial f}{\partial \boldsymbol{\omega}}(\bar{\mathbf{P}}, \mathbf{t}, \mathtt{R}) &= d(\pi) \times \frac{\partial}{\partial \boldsymbol{\omega}} \left(\mathtt{K} \left(\mathtt{R}_0(\boldsymbol{\omega}) \right)^\top [\mathtt{I}_3 | -\mathbf{t}] \, \bar{\mathbf{P}} \right) \\ &= d(\pi) \times \mathtt{K} \times \frac{\partial}{\partial \boldsymbol{\omega}} \left((\mathtt{R}(\boldsymbol{\omega})\mathtt{R}_0)^\top \right) [\mathtt{I}_3 | -\mathbf{t}] \, \bar{\mathbf{P}} \\ &= d(\pi) \times \mathtt{K}\mathtt{R}_0^\top \times -\frac{\partial}{\partial \boldsymbol{\omega}} \left(\mathtt{R}(\boldsymbol{\omega}) \right) [\mathtt{I}_3 | -\mathbf{t}] \, \bar{\mathbf{P}} \end{aligned}$$

Avec l'hypothèse des petis angles, chaque dérivée s'écrit :

$$\frac{\partial f}{\partial \omega_j}(\bar{\mathbf{P}}, \mathbf{t}, \mathtt{R}) = \underbrace{d(\pi) \times \mathtt{K}\mathtt{R}_0^\top \times -\frac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_j} \times [\mathtt{I}_3 | -\mathbf{t}] \, \bar{\mathbf{P}}}_{(2\times3)(3\times3)(3\times3)(3\times3)(3\times4)(4\times1)=(2\times1) \ (\times \mathbf{3})}$$

où les $\dfrac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_j}$, $j \in \{1, 2, 3\}$ sont :

$$\frac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_1} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \frac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_2} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \text{ et } \frac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_3} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Cas générique : erreur angulaire

Comme nous l'avons expliqué dans la section 4.3, l'ajustement de faisceaux générique minimise une erreur angulaire entre les directions des rayons optiques $\mathbf{d}$ et $\mathbf{D}$ qui s'exprime sous la forme :

$$\boldsymbol{\epsilon} = \pi(\mathtt{R}'\mathbf{D}) = \pi \left(\mathtt{R}' \left(\mathtt{R}^\top [\mathtt{I}_3 \mid -\mathbf{t}] \bar{\mathbf{P}} - T\mathbf{s} \right) \right)$$

avec $\mathtt{R}'$ matrice de rotation telle que $\mathtt{R}'\mathbf{d} = [0 \ 0 \ 1]^\top$, et $\mathbf{s}$ le point de départ des deux rayons optiques dans le repère de la caméra.
De la même manière que pour l'erreur de reprojection image, on a

$$d(f) = d(\pi) \times \mathtt{R}' \times d \left(\mathtt{R}^\top [\mathtt{I}_3 \mid -\mathbf{t}] \bar{\mathbf{P}} - T\mathbf{s} \right)$$

et les dérivées s'obtiennent de façon très similaire.

Dérivées par rapport aux paramètres des points 3D

$$\begin{aligned}\frac{\partial f}{\partial \bar{\mathbf{P}}}(\bar{\mathbf{P}},\mathbf{t},\mathtt{R}) &= d(\pi) \times \mathtt{R}' \times \frac{\partial}{\partial \bar{\mathbf{P}}}\left(\mathtt{R}^\top[\mathtt{I}_3 \mid -\mathbf{t}]\bar{\mathbf{P}} - T\mathbf{s}\right)\\ &= \underbrace{d(\pi) \times \mathtt{R}' \times \left(\mathtt{R}^\top[\mathtt{I}_3 \mid -\mathbf{t}] - [\mathtt{0}_{3\times 3} \mid \mathbf{s}]\right)}_{(2\times3)(3\times3)(3\times4)=(2\times4)}\end{aligned}$$

Dérivées par rapport aux paramètres des caméras

1. Dérivées par rapport aux paramètres de translation :

$$\begin{aligned}\frac{\partial f}{\partial \mathbf{t}}(\bar{\mathbf{P}},\mathbf{t},\mathtt{R}) &= d(\pi) \times \mathtt{R}' \times \frac{\partial}{\partial \mathbf{t}}\left(\mathtt{R}^\top[\mathtt{I}_3 \mid -\mathbf{t}]\bar{\mathbf{P}} - T\mathbf{s}\right)\\ &= -T \times \underbrace{d(\pi) \times \mathtt{R}'\mathtt{R}^\top}_{(2\times3)(3\times3)(3\times3)=(2\times3)}\end{aligned}$$

2. Dérivées par rapport aux paramètres de rotation :
   On utilise le même paramétrage que dans le cas spécifique.

$$\begin{aligned}\frac{\partial f}{\partial \boldsymbol{\omega}}(\bar{\mathbf{P}},\mathbf{t},\mathtt{R}) &= d(\pi) \times \mathtt{R}' \times \frac{\partial}{\partial \boldsymbol{\omega}}\left(\left(\mathtt{R}_0(\boldsymbol{\omega})\right)^\top[\mathtt{I}_3 \mid -\mathbf{t}]\bar{\mathbf{P}} - T\mathbf{s}\right)\\ &= d(\pi) \times \mathtt{R}' \times \frac{\partial}{\partial \boldsymbol{\omega}}\left(\left(\mathtt{R}(\boldsymbol{\omega})\mathtt{R}_0\right)^\top\right)[\mathtt{I}_3 \mid -\mathbf{t}]\bar{\mathbf{P}}\\ &= d(\pi) \times \mathtt{R}'\mathtt{R}_0^\top \times -\frac{\partial}{\partial \boldsymbol{\omega}}\left(\mathtt{R}(\boldsymbol{\omega})\right)[\mathtt{I}_3 \mid -\mathbf{t}]\bar{\mathbf{P}}\end{aligned}$$

   On obtient alors chaque dérivée en faisant :

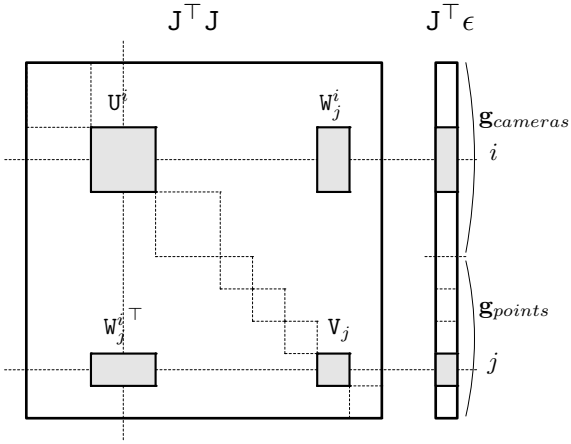
$$\frac{\partial f}{\partial \omega_j}(\bar{\mathbf{P}},\mathbf{t},\mathtt{R}) = \underbrace{d(\pi) \times \mathtt{R}'\mathtt{R}_0^\top \times -\frac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_j} \times [\mathtt{I}_3 | -\mathbf{t}]\,\bar{\mathbf{P}}}_{(2\times3)(3\times3)(3\times3)(3\times3)(3\times4)(4\times1)=(2\times1)\ \textbf{(}\times\textbf{3)}}$$

   avec les $\dfrac{\partial [\boldsymbol{\omega}]_\times}{\partial \omega_j}$ comme définis précédemment.

Construction et résolution du système creux

Les dérivées permettent de calculer la matrice Jacobienne $\mathtt{J}$ de la fonction de projection et ainsi de construire le système des équations normales. Cependant, l'évaluation de la matrice $\mathtt{J}$ complète n'est pas indispensable et on peut calculer directement $\mathtt{J}^\top\mathtt{J}$ et $\mathtt{J}^\top\boldsymbol{\epsilon}$. On construit incrémentalement le

système en parcourant toutes les observations : l'observation du j-ème par la i-ème caméra fait intervenir les blocs $\mathtt{U}^i$, $\mathtt{V}_j$ et $\mathtt{W}^i_j$ de la matrice $\mathtt{J}^\top\mathtt{J}$ ainsi que le i-ème bloc de $\mathbf{g}_{cameras}$ et le j-ème bloc de $\mathbf{g}_{points}$.



Au départ, tous les blocs sont initialisés à zéro. Ensuite, ils sont mis à jour pour chaque observation (i, j) :

$$
\begin{aligned}
&\cdot\ \mathtt{U}^i = \mathtt{U}^i + \left(\frac{\partial f^i_j}{\partial C^i}\right)^\top \left(\frac{\partial f^i_j}{\partial C^i}\right)\\
&\cdot\ \mathtt{V}_j = \mathtt{V}_j + \left(\frac{\partial f^i_j}{\partial \mathbf{P}_j}\right)^\top \left(\frac{\partial f^i_j}{\partial \mathbf{P}_j}\right)\\
&\cdot\ \mathtt{W}^i_j = \left(\frac{\partial f^i_j}{\partial C^i}\right)^\top \left(\frac{\partial f^i_j}{\partial \mathbf{P}_j}\right)\\
&\cdot\ \mathbf{g}_{cameras}(i) = \mathbf{g}_{cameras}(i) + \left(\frac{\partial f^i_j}{\partial C^i}\right)^\top \epsilon^i_j\\
&\cdot\ \mathbf{g}_{points}(j) = \mathbf{g}_{points}(j) + \left(\frac{\partial f^i_j}{\partial \mathbf{P}_j}\right)^\top \epsilon^i_j
\end{aligned}
$$

Le système obtenu a la forme suivante :

$$\begin{bmatrix} \mathtt{U} & \mathtt{W} \\ \mathtt{W}^\top & \mathtt{V} \end{bmatrix} \begin{bmatrix} \mathbf{\Delta}_{cameras} \\ \mathbf{\Delta}_{points} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{cameras} \\ \mathbf{g}_{points} \end{bmatrix}$$

La première étape de la résolution consiste à remarquer qu'en multipliant les deux membres à gauche par $\begin{bmatrix} \mathtt{I} & -\mathtt{W}\mathtt{V}^{-1} \\ \mathtt{0} & \mathtt{I} \end{bmatrix}$, on élimine le bloc en haut à droite de la première matrice, ce qui donne :

$$\begin{bmatrix} \mathtt{U} - \mathtt{W}\mathtt{V}^{-1}\mathtt{W}^\top & \mathtt{0} \\ \mathtt{W}^\top & \mathtt{V} \end{bmatrix} \begin{bmatrix} \mathbf{\Delta}_{cameras} \\ \mathbf{\Delta}_{points} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{cameras} - \mathtt{W}\mathtt{V}^{-1}\mathbf{g}_{points} \\ \mathbf{g}_{points} \end{bmatrix}$$

La résolution se fait alors en deux étapes :

1. Le calcul du pas $\mathbf{\Delta}_{cameras}$ qui doit être appliqué aux paramètres des caméras par résolution du système linéaire suivant :

$$(\mathtt{U} - \mathtt{W}\mathtt{V}^{-1}\mathtt{W}^{\top})\mathbf{\Delta}_{cameras} = \mathbf{g}_{cameras} - \mathtt{W}\mathtt{V}^{-1}\mathbf{g}_{points}$$

   La matrice $\mathtt{V}$ est facilement inversible car elle est diagonale par blocs de taille 3×3. En effet, une normalisation est appliquée sur les coordonnées homogènes des points 3D. Elle consiste à fixer la coordonnée de plus grande valeur absolue à 1 et de calculer uniquement les dérivées par rapport aux 3 autres coordonnées. La résolution du système se fait par décomposition de la matrice $\mathtt{U} - \mathtt{W}\mathtt{V}^{-1}\mathtt{W}^{\top}$ par la méthode de CHOLESKY suivie de deux résolutions de systèmes linéaires triangulaires.

2. La deuxième étape est le calcul direct du pas $\mathbf{\Delta}_{points}$ applicable aux points 3D :

$$\mathbf{\Delta}_{points} = \mathtt{V}^{-1}(\mathbf{g}_{points} - \mathtt{W}^{\top}\mathbf{\Delta}_{cameras})$$

   Le calcul est immédiat à partir de $\mathbf{\Delta}_{cameras}$ calculé à l'étape précédente.

Spécificités de l'ajustement de faisceaux local

Lors de l'ajustement de faisceaux local, les paramètres optimisés sont : n poses de caméras et les m points 3D visibles dans les n images correspondantes. Cela signifie que les matrices $\mathtt{U}$, $\mathtt{V}$ et $\mathtt{W}$ sont réduites et sont respectivement de taille $6n \times 6n$, $3m \times 3m$ et $6n \times 3m$.
Le critère de reprojection est minimisé sur une séquence de N images : en plus des observations des m points dans les n images, on intègre donc les observations des points dans $N - m$ images supplémentaires. Chaque observation supplémentaire (i, j) (la caméra i n'est pas optimisée) se traduit uniquement par une nouvelle contrainte sur les blocs $\mathtt{V}_j$ et $\mathbf{g}_{points}(j)$ après avoir calculé les dérivées $\dfrac{\partial f_j^i}{\partial \mathbf{P}_j}$.

Annexe 2 : solutions exactes à l'équation (linéaire) de Pless

Dans cette annexe, on exhibe des solutions à l'équation (linéaire) de Pless qui ne sont pas la solution que l'on recherche. Il est possible qu'il y en ait d'autres.

La i-ème paire de rayons est définie par les coordonnées de Plücker $(\mathbf{q}_{0i}, \mathbf{q}'_{0i})$ et $(\mathbf{q}_{1i}, \mathbf{q}'_{1i})$ telles que $\mathbf{q}'_{0i} = \mathbf{q}_{0i} \wedge \mathbf{c}_{0i} = -[\mathbf{c}_{0i}]_\times \mathbf{q}_{0i}$ et $\mathbf{q}'_{1i} = \mathbf{q}_{1i} \wedge \mathbf{c}_{1i} = -[\mathbf{c}_{1i}]_\times \mathbf{q}_{1i}$. Les rayons de la i-ème paire s'intersectent s'ils satisfont l'équation Pless (ou plus exactement, sa version linéaire)

$$\begin{aligned} 0 &= \mathbf{q}'^{\top}_{0i} \tilde{\mathtt{R}} \mathbf{q}_{1i} - \mathbf{q}^{\top}_{0i} \tilde{\mathtt{E}} \mathbf{q}_{1i} + \mathbf{q}^{\top}_{0i} \tilde{\mathtt{R}} \mathbf{q}'_{1i} \\ &= \mathbf{q}^{\top}_{0i} ([\mathbf{c}_{0i}]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{E}} - \tilde{\mathtt{R}} [\mathbf{c}_{1i}]_\times) \mathbf{q}_{1i} \end{aligned}$$

avec les deux matrices $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}})$ de taille 3×3 comme inconnues. Dans cette section, on souhaite trouver $(\tilde{\mathtt{R}}, \tilde{\mathtt{E}})$ tel que $\forall i, \tilde{\mathtt{E}} = [\mathbf{c}_{0i}]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}} [\mathbf{c}_{1i}]_\times$. On recherche alors $\tilde{\mathtt{R}} \in \mathbb{R}^{3\times 3}$ tel que $\tilde{\mathtt{E}}$ soit indépendant de toutes les paires de centres possibles $(\mathbf{c}_{0i}, \mathbf{c}_{1i})$. La conséquence est que l'équation 4.13 est exactement égale à 0 (à la précision de la machine près) pour tout $(\mathbf{q}_{0i}, \mathbf{q}_{1i})$. On considère plusieurs cas.

Cas 1 : appariements simples $(\forall i, \mathbf{c}_{0i} = \mathbf{c}_{1i})$

Comme cela a été dit dans la section 4.4.2, ce cas particulier est important en pratique.

Caméra Centrale Soit $\mathbf{c}$ le centre. Ce cas est simple : on a $\mathbf{c}_{0i} = \mathbf{c}_{1i} = \mathbf{c}$ et $\tilde{\mathtt{E}} = [\mathbf{c}]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}} [\mathbf{c}]_\times$. Tout $\tilde{\mathtt{R}} \in \mathbb{R}^{3\times 3}$ est possible.

Paire stéréo Soient $\mathbf{c}_a$ et $\mathbf{c}_b$ les deux centres. On a $(\mathbf{c}_{0i}, \mathbf{c}_{1i}) \in \{(\mathbf{c}_a, \mathbf{c}_a), (\mathbf{c}_b, \mathbf{c}_b)\}$ et $\tilde{\mathtt{E}} = [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}} [\mathbf{c}_a]_\times = [\mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}} [\mathbf{c}_b]_\times$. La contrainte sur $\tilde{\mathtt{R}}$ est donc

$$[\mathbf{c}_a - \mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}} [\mathbf{c}_a - \mathbf{c}_b]_\times = 0.$$

Tout $\tilde{\mathtt{R}}$ dans l'espace vectoriel des polynômes en $[\mathbf{c}_a - \mathbf{c}_b]_\times$ est possible. De plus, il est simple de prouver que cette contrainte ne permet pas d'autres $\tilde{\mathtt{R}}$ en changeant le système de coordonnées de sorte que $\mathbf{c}_a - \mathbf{c}_b = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top$. On en déduit que

$$\begin{aligned} &\tilde{\mathtt{E}} = [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}} [\mathbf{c}_a]_\times \text{ et} \\ &\tilde{\mathtt{R}} \in Vect\{\mathtt{I}_{3\times 3}, [\mathbf{c}_a - \mathbf{c}_b]_\times, (\mathbf{c}_a - \mathbf{c}_b)(\mathbf{c}_a - \mathbf{c}_b)^\top\}. \end{aligned}$$

Caméra axiale Tous les centres de la caméra sont alignés : il existe $\mathbf{c}_a$ et $\mathbf{c}_b$ tels que $\mathbf{c}_{0i} = \mathbf{c}_{1i} = (1-\lambda_i)\mathbf{c}_a + \lambda_i \mathbf{c}_b$ avec $\lambda_i \in \mathbb{R}$. Donc,

$$\begin{aligned}\tilde{\mathtt{E}}(\lambda) &= [(1-\lambda)\mathbf{c}_a + \lambda\mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[(1-\lambda)\mathbf{c}_a + \lambda\mathbf{c}_b]_\times \\ &= [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_a]_\times + \lambda([\mathbf{c}_b - \mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_b - \mathbf{c}_a]_\times)\end{aligned}$$

ne doit pas dépendre de λ. On voit que $[\mathbf{c}_b - \mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_b - \mathbf{c}_a]_\times = 0$. Ce cas est le même que celui qui précède.

Caméra non axiale Il existe alors trois centres $\mathbf{c}_a, \mathbf{c}_b, \mathbf{c}_c$ non alignés. Maintenant, la contrainte sur $\tilde{\mathtt{R}}$ est

$$\begin{aligned}0 &= [\mathbf{c}_a - \mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_a - \mathbf{c}_b]_\times \\ &= [\mathbf{c}_b - \mathbf{c}_c]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_b - \mathbf{c}_c]_\times \\ &= [\mathbf{c}_c - \mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_c - \mathbf{c}_a]_\times .\end{aligned}$$

Cette contrainte est trois fois celle de la paire stéréo. Changeons le système de coordonnées de sorte que $\{\mathbf{c}_a - \mathbf{c}_b, \mathbf{c}_b - \mathbf{c}_c, \mathbf{c}_c - \mathbf{c}_a\}$ soit la base canonique et écrivons les solutions des trois cas stéréo. On voit alors que seul $\tilde{\mathtt{R}} \in Vect\{\mathtt{I}_{3\times 3}\}$ est possible.

Cas 2 : appariements quelconques

Maintenant, les appariements sont simples ou pas.

Caméra Centrale Ce cas ne peut se produire ici.

Paire stéréo Dans ce cas, on a $(\mathbf{c}_{0i}, \mathbf{c}_{1i}) \in \{(\mathbf{c}_a, \mathbf{c}_a), (\mathbf{c}_b, \mathbf{c}_b), (\mathbf{c}_a, \mathbf{c}_b), (\mathbf{c}_b, \mathbf{c}_a)\}$ et $\tilde{\mathtt{E}} = [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_a]_\times = [\mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_b]_\times = [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_b]_\times = [\mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_a]_\times$. La contrainte sur $\tilde{\mathtt{R}}$ est donc

$$0 = [\mathbf{c}_a - \mathbf{c}_b]_\times \tilde{\mathtt{R}} = \tilde{\mathtt{R}}[\mathbf{c}_a - \mathbf{c}_b]_\times .$$

Cette contrainte est plus forte que dans le cas de la caméra stéréo avec des appariements simples. On voit alors que seul $\tilde{\mathtt{R}} \in Vect\{(\mathbf{c}_a - \mathbf{c}_b)(\mathbf{c}_a - \mathbf{c}_b)^\top\}$ est possible.

Caméra axiale Il existe $\mathbf{c}_a$ et $\mathbf{c}_b$ tels que $\mathbf{c}_{0i} = (1-\lambda_{0i})\mathbf{c}_a + \lambda_{0i}\mathbf{c}_b$ et $\mathbf{c}_{1i} = (1-\lambda_{1i})\mathbf{c}_a + \lambda_{1i}\mathbf{c}_b$ avec $\lambda_{0i} \in \mathbb{R}$ et $\lambda_{1i} \in \mathbb{R}$. Donc,

$$\begin{aligned}\tilde{\mathtt{E}}(\lambda_0, \lambda_1) &= [(1-\lambda_0)\mathbf{c}_a + \lambda_0\mathbf{c}_b]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[(1-\lambda_1)\mathbf{c}_a + \lambda_1\mathbf{c}_b]_\times \\ &= [\mathbf{c}_a]_\times \tilde{\mathtt{R}} - \tilde{\mathtt{R}}[\mathbf{c}_a]_\times + \lambda_0[\mathbf{c}_b - \mathbf{c}_a]_\times \tilde{\mathtt{R}} - \lambda_1 \tilde{\mathtt{R}}[\mathbf{c}_b - \mathbf{c}_a]_\times\end{aligned}$$

ne doit pas dépendre de λ_0 et λ_1 : on a $0 = [\mathbf{c}_b - \mathbf{c}_a]_\times \tilde{\mathtt{R}} = \tilde{\mathtt{R}}[\mathbf{c}_b - \mathbf{c}_a]_\times$. Ce cas est le même que le précédent.

Caméra non axiale Il existe trois centres non alignés $\mathbf{c}_a, \mathbf{c}_b, \mathbf{c}_c$ et la contrainte sur $\tilde{\mathtt{R}}$ est trois fois celle du cas stéréo :

$$\begin{aligned}0 &= [\mathbf{c}_a - \mathbf{c}_b]_\times \tilde{\mathtt{R}} = \tilde{\mathtt{R}}[\mathbf{c}_a - \mathbf{c}_b]_\times = [\mathbf{c}_b - \mathbf{c}_c]_\times \tilde{\mathtt{R}} \\ &= \tilde{\mathtt{R}}[\mathbf{c}_b - \mathbf{c}_c]_\times = [\mathbf{c}_c - \mathbf{c}_a]_\times \tilde{\mathtt{R}} = \tilde{\mathtt{R}}[\mathbf{c}_c - \mathbf{c}_a]_\times.\end{aligned}$$

Changeons le système de coordonnées de sorte que $\{\mathbf{c}_a - \mathbf{c}_b, \mathbf{c}_b - \mathbf{c}_c, \mathbf{c}_c - \mathbf{c}_a\}$ soit la base canonique et écrivons les solutions des trois cas stéréo induits. On obtient $\tilde{\mathtt{R}} = 0$. Aucune solution exacte n'est obtenue avec cette méthode dans ce cas.

Bibliographie

[1] C. Baillard et A. Zisserman. Automatic reconstruction of piecewise planar models from multiple views. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 1999.

[2] A. Bartoli et P. Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *International Journal on Computer Vision*, 52(1) :45–64, 2003.

[3] A. Bartoli et P. Sturm. Structure-From-Motion using lines : Representation, triangulation and bundle adjustment. *Computer Vision and Image Understanding*, 100(3) :416–441, 2005.

[4] H. Bay, T. Tuytelaars, et L. Van Gool. SURF : Speeded up robust features. In *Proc. of European Conference on Computer Vision*, 2006.

[5] P. Beardsley, P. Torr, et A. Zisserman. 3D model acquisition from extended image sequences. In *Proc. of European Conference on Computer Vision*, 1996.

[6] P.R. Beaudet. Rotationally invariant image operators. In *Proc. of International Conference on Pattern Recognition*, pages 579–583, 1978.

[7] P. Chang et M. Hebert. Omni-directional structure from motion. In *Proc. of the 2000 IEEE Workshop on Omnidirectional Vision*, pages 127–133, 2000.

[8] A. Chiuso, P. Favaro, H. Jin, et S. Soatto. MFm : 3-D motion from 2-D motion causally integrated over time. In *Proc. of European Conference on Computer Vision*, volume 1842, pages 735–750, 2000.

[9] L. A. Clemente, A. J. Davison, I. Reid, J. Neira, et J. D. Tardós. Mapping large loops with a single hand-held camera. In *Proc. of Robotics : Science and Systems Conference*, 2007.

[10] K. Cornelis, F. Verbiest, et L.J. Van Gool. Drift detection and removal for sequential structure from motion algorithms. *Trans. on*

Pattern Analysis and Machine Intelligence, 26(10) :1249–1259, October 2004.

[11] S. Cornou, M. Dhome, et P. Sayd. Architectural reconstruction with multiple views and geometric constraints. In *Proc. of British Machine Vision Conference*, 2003.

[12] A. Davison. Real-time simultaneous localization and mapping with a single camera. In *Proc. of International Conference on Computer Vision*, 2003.

[13] M. Deans et M. Hebert. Experimental comparison of techniques for localization and mapping using a bearings only sensor. In *Proc. of the ISER '00 Seventh International Symposium on Experimental Robotics*, 2000.

[14] D. F. DeMenthon et L. S. Davis. Model-based object pose in 25 lines of code. In *Proc. of European Conference on Computer Vision*, pages 335–343, 1992.

[15] M. Dhome, M. Richetin, J.T. Lapresté, et G. Rives. Determination of the attitude of 3-d objects from a single perspective view. *Trans. on Pattern Analysis and Machine Intelligence*, 11(12) :1265–1278, December 1989.

[16] L. Dreschler et H. H. Nagel. Volumetric model and 3d-trajectory of a moving car derived from monocular tv-frame sequences of a street scene. In *Proc. of the 7th International Joint Conferences on Artificial Intelligence*, pages 692–697, 1981.

[17] C. Engels, H. Stewénius, et D. Nistér. Bundle adjustment rules. In *Photogrammetric Computer Vision (PCV)*, August 2006.

[18] A. Eudes. Utilisation de descripteurs pour l'odométrie visuelle temps-réel. Master's thesis, MASTER STIC-VIRO, Université Blaise Pascal de Clermont-Ferrand, 2007.

[19] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. MIT Press, 1993.

[20] O. Faugeras et M. Hebert. The representation, recognition, and locating of 3-d objects. *International Journal of Robotic Research*, 5(3) :27–52, 1986.

[21] M. A. Fischler et R. C. Bolles. Random sample consensus, a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6) :381 – 395, 1981.

[22] A. W. Fitzgibbon et A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proc. of European Conference on Computer Vision*, 1998.

[23] J. Grunert. Das pothenotische problem in erweiterter gestalt nebst ueber seineanwendungen in der geodasie. *Grunerts Archiv für Mathematik und Physic*, (1) :238–248, 1841.

[24] R. M. Haralick, C-N Lee, K. Ottenberg, et M. Noelle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal on Computer Vision*, 13(3) :331–356, 1994.

[25] C. Harris et M. Stephens. A combined corner and edge detector. In *4th ALVEY Vision Conference*, pages 147–151, 1988.

[26] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In *Applications of Invariance in Computer Vision*, pages 237–256, 1993.

[27] R. I. Hartley. In defence of the 8-point algorithm. In *Proc. of International Conference on Computer Vision*, page 1064, 1995.

[28] R. I. Hartley. Lines and points in three views and the trifocal tensor. *International Journal on Computer Vision*, 22(2) :125–140, 1997.

[29] R. I. Hartley et P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2) :146–157, 1997.

[30] R. I. Hartley et A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN : 0521623049, 2000.

[31] P.V.C Hough. Machine analysis of bubble chamber pictures. In *International Conference on High Energy Accelerators ans Instrumentation*, pages 554–556, 1959.

[32] P. J. Huber. *Robust statistics*. J. Wiley & Sons, 1981.

[33] S.B. Kang et R. Szeliski. 3D scene data recovery using omnidirectional multibaseline stereo. *International Journal on Computer Vision*, 25(2) :167–183, 1997.

[34] J. Kannala et S.S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *Trans. on Pattern Analysis and Machine Intelligence*, 28(8) :1335–1340, 2006.

[35] Y. Ke et R. Sukthankar. PCA-SIFT : A more distinctive representation for local image descriptors. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 2004.

[36] J. Kim, J. S. Kim, K. J. Yoon, et I. Kweon. Visual slam by single-camera catadioptric stereo. In *SICE-ICASE International Joint Conference*, 2006.

[37] J.H. Kim et S. Sukkarieh. Airborne simultaneous localisation and map building. In *Proc. of International Conference on Robotics and Automation*, pages 406–411, 2003.

[38] J.M. Lavest, M. Viala, et M. Dhome. Do we need an accurate calibration pattern to achieve a reliable camera calibration ? In *Proc. of European Conference on Computer Vision*, pages 158–174, 1998.

[39] T. Lemaire et S. Lacroix. SLAM with Panoramic Vision. *Journal for Field Robotics, special issue SLAM in the Fields*, 2006.

[40] J. J. Leonard et H. F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *Proc of International Conference on Intelligent Robots and Systems, workshop on Intelligence for Mechanical Systems*, volume 3, pages 1442–1447, 1991.

[41] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.*, 2 :164–168, 1944.

[42] L. Leyrit. Localisation par vision monoculaire pour les créneaux automatiques. Master's thesis, MASTER STIC-VIRO, Université Blaise Pascal de Clermont-Ferrand, 2006.

[43] M. Lhuillier. Robust dense matching using local and global geometric constraints. In *Proc. of International Conference on Pattern Recognition*, volume 2, pages 968–972, 2000.

[44] M. Lhuillier. Automatic scene structure and camera motion using a catadioptric system. *Computer Vision and Image Understanding*, 2007. doi :10.1016/j.cviu.2007.05.004.

[45] M. Lhuillier et L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *Trans. on Pattern Analysis and Machine Intelligence*, 27(3) :418–433, 2005.

[46] R. Li, K. Di, J. Wang, S. Agarwal, L. Matthies, A. Howard, et R. Willson. Incremental bundle adjustment techniques using networked overhead and ground imagery for long-range autonomous mars rover localization. In *Proc. of the iSAIRA Conference*, 2005.

[47] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293 :133–135, 1981.

[48] M.I.A. Lourakis et A.A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Technical report, Institute of Computer Science - FORTH, Heraklion, Greece, Institute of Computer Science - FORTH, Heraklion, Greece, 2004. Available at ftp ://ftp.ics.forth.gr/tech-reports/2004.

[49] M.I.A. Lourakis et A.A. Argyros. Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment ? In *Proc. of International Conference on Computer Vision*, pages II : 1526–1531, 2005.

[50] D. Lowe. Fitting parameterized three-dimensional models to images. *Trans. on Pattern Analysis and Machine Intelligence*, 13(5) :441–450, 1991.

[51] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision*, 20 :91–110, 2003.

[52] C. P. Lu, G. D. Hager, et E. Mjolsness. Fast and globally convergent pose estimation from video images. *Trans. on Pattern Analysis and Machine Intelligence*, 22(6) :610–622, 2000.

[53] K. Madsen, H. B. Nielsen, et O. Tingleff. *Methods for Non-Linear Least Squares Problems (2nd ed.)*. Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2004.

[54] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Industr. Appl. Math.*, 11(1) :431–444, 1963.

[55] D. Martinec et T. Pajdla. Line reconstruction from many perspective images by factorization. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 2003.

[56] H. Martinsson, F. Gaspard, A. Bartoli, et J.-M. Lavest. Energy-based reconstruction of 3D curves for quality control. In *Proceedings of the Sixth IAPR International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2007.

[57] P. McLauchlan. A batch/recursive algorithm for 3D scene reconstruction. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 2000.

[58] B. Micusik et T. Pajdla. Autocalibration & 3D reconstruction with non-central catadioptric cameras. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 2004.

[59] K. Mikolajczyk et C. Schmid. An affine invariant interest point detector. In *Proc. of European Conference on Computer Vision*, pages 128–142, 2002.

[60] K. Mikolajczyk et C. Schmid. A performance evaluation of local descriptors. *Trans. on Pattern Analysis and Machine Intelligence*, 27(10) :1615–1630, 2005.

[61] M. Montemerlo, S. Thrun, D. Koller, et B. Wegbreit. Fastslam : a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, 2002.

[62] H. P. Moravec. Rover visual obstacle avoidance. In *Proc. of the 7th International Joint Conference on Artificial Intellignece*, pages 785–790, 1981.

[63] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, et P. Sayd. 3D reconstruction of complex structures with bundle adjustment : an incremental approach. In *Proc. of International Conference on Robotics and Automation*, May 2006.

[64] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, et P. Sayd. Monocular vision based SLAM for mobile robots. In *Proc. of International Conference on Pattern Recognition*, 2006.

[65] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, et P. Sayd. Real time localization and 3D reconstruction. In *Proc. of Conference on Computer Vision and Pattern Recognition*, June 2006.

[66] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, et P. Sayd. Generic and real-time structure from motion. In *Proc. of British Machine Vision Conference*, 2007.

[67] P. M. Newman et J. J. Leonard. Consistent convergent constant time SLAM. In *International Joint Conference on Artificial Intelligence*, 2003.

[68] D. Nister. *Automatic Dense Reconstruction from Uncalibrated Video Sequences*. PhD thèse, Royal Institute of Technology (KTH), Stockholm, Sweden, 2001.

[69] D. Nister. An efficient solution to the five-point relative pose problem. *Trans. on Pattern Analysis and Machine Intelligence*, 26(6) :756–777, 2004.

[70] D. Nister, O. Naroditsky, et J. Bergen. Visual odometry. In *Proc. of Conference on Computer Vision and Pattern Recognition*, pages 652–659, 2004.

[71] R. Pless. Using many cameras as one. In *Proc. of Conference on Computer Vision and Pattern Recognition*, pages II : 587–593, 2003.

[72] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, et R. Koch. Visual modeling with a hand-held camera. *International Journal on Computer Vision*, 59(3) :207–232, 2004.

[73] M. Pollefeys, R. Koch, M. Vergauwen, et L. Van Gool. Automated reconstruction of 3D scenes from sequences of images. *Isprs Journal Of Photogrammetry And Remote Sensing*, 55(4) :251–267, 2000.

[74] M. J. D. Powell. A hybrid method for nonlinear equations. *Numerical methods for nonlinear algebraic equations*, pages 87–161, 1970.

[75] W. H. Press, Saul A. Teukolsky, W. T. Vetterling, et Brian P. Flannery. *Numerical recipes in C*. Cambridge University Press, second edition, 1992. The art of scientific computing.

[76] L. Quan et Z. Lan. Linear n >=4-point pose determination. In *Proc. of International Conference on Computer Vision*, pages 778–783, 1998.

[77] S. Ramalingam, S. Lodha, et P. Sturm. A generic structure-from-motion framework. *Computer Vision and Image Understanding*, 103(3) :218–228, 2006.

[78] J. Repko et M. Pollefeys. 3D models from extended uncalibrated video sequences : Addressing key-frame selection and projective drift. In *Proc. of International Conference on 3D Digital Imaging and Modeling*, 2005.

[79] E. Rosten et T. Drummond. Machine learning for high-speed corner detection. In *Proc. of European Conference on Computer Vision*, pages 430–443, 2006.

[80] P. J. Rousseeuw. Least median of squares regression. *Journal American Statistic Association*, 79(388) :871–880, 1984.

[81] P. J. Rousseeuw et A. M. Leroy. *Robust regression and outlier detection*. J. Wiley & Sons, 1987.

[82] E. Royer. *Cartographie 3D et localisation par vision monoculaire pour la navigation autonome d'un robot mobile*. PhD thèse, LASMEA, Université Blaise Pascal de Clermont-Ferrand, 2006.

[83] E. Royer, M. Lhuillier, M. Dhome, et T. Chateau. Localization in urban environments : monocular vision compared to a differential gps sensor. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 2005.

[84] E. Royer, M. Lhuillier, M. Dhome, et J. M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision (special joint issue on vision and robotics, with the International Journal of Robotics Research)*, 74(3), 2007.

[85] M. Sarkis, K. Diepold, et K. Hüper. A fast and robust solution to the five-point relative pose problem using gauss-newton optimization on a manifold. In *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, 2007.

[86] D. Scharstein et R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal on Computer Vision*, 47(1) :7–42, 2002.

[87] C. Schmid, R. Mohr, et C. Bauckhage. Comparing and evaluating interest points. In *Proc. of International Conference on Computer Vision*, 1998.

[88] C. Schmid et A. Zisserman. Automatic line matching across views. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 1997.

[89] S. Se, D. Lowe, et J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8) :735–758, 2002.

[90] J. Shi et C. Tomasi. Good features to track. In *Proc. of Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[91] H. Y. Shum, Q. Ke, et Z. Zhang. Efficient bundle adjustment with virtual key frames : A hierarchical approach to multi-frame structure from motion. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 1999.

[92] R. Sim, P. Elinas, et J.J. Little. A study of the rao-blackwellised particle filter for efficient and accurate vision-based slam. *International Journal on Computer Vision*, 74(3) :303–318, 2007.

[93] N. Simond et P. Rives. Détection robuste du plan de la route en milieu urbain. In *14ème Congrès Francophone de Reconnaissance des Formes et Intelligence Artificielle (RFIA'04)*, 2004.

[94] R. C. Smith et P. Cheeseman. On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4) :56–68, 1986.

[95] D. Steedly et I. Essa. Propagation of innovative information in Non-Linear Least-Squares structure from motion. In *Proc. of International Conference on Computer Vision*, pages 223–229, 2001.

[96] H. Stewénius, D. Nistér, M. Oskarsson, et K. Åström. Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision*, 2005.

[97] H. Strasdat, C. Stachniss, M. Bennewitz, et W. Burgard. Visual bearing-only simultaneous localization and mapping with improved feature matching. In *Autonome Mobile Systeme (AMS)*, Kaiserslautern, Germany, 2007.

[98] P. Sturm et S. Ramalingam. A generic concept for camera calibration. In *Proceedings of the European Conference on Computer Vision, Prague, Czech Republic*, volume 2, pages 1–13, 2004.

[99] T. Thormählen, H. Broszio, et A. Weissenfeld. Keyframe selection for camera motion and structure estimation from multiple views. In *Proc. of European Conference on Computer Vision*, 2004.

[100] C. Tomasi, J. Zhang, et D. Redkey. Experiments with a real-time structure-from-motion system. In *Proc. IV International Symp Experimental Robotics*, 1995.

[101] P.H.S. Torr et D.W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal on Computer Vision*, 24(3) :271–300, 1997.

[102] B. Triggs, P. McLauchlan, R. Hartley, et A. Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, et R. Szeliski, editors, *Vision Algorithms : Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[103] J. W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[104] J.S.C Yuan. A general photogrammetric solution for the determining object position and orientation. *Trans. on Robotics and Automation*, 5(2) :129–142, 1989.

[105] G. Zhang, X. Qin, W. Hua, T. T. Wong, P. A. Heng, et H. Bao. Robust metric reconstruction from challenging video sequences. In *Proc. of Conference on Computer Vision and Pattern Recognition*, 2007.

[106] R. Zhang, P.S. Tsai, J.E. Cryer, et M. Shah. Shape from shading : A survey. *Trans. on Pattern Analysis and Machine Intelligence*, 21(8) :690–706, 1999.

[107] Z. Zhang, R. Deriche, O. D. Faugeras, et Q. T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1/2) :87–119, 1995.

[108] Z. Zhang et Y. Shan. Incremental motion estimation through modified bundle adjustment. In *Proc. of International Conference on Image Processing*, pages II : 343–346, 2003.

Reconstruction 3D et localisation simultanée de caméras mobiles : une approche temps-réel par ajustement de faisceaux local

Le problème de la reconstruction 3D à partir d'une séquence d'images acquise par une caméra en mouvement est un sujet important dans le domaine de la vision par ordinateur. Ce travail de thèse présente une méthode qui permet d'estimer conjointement des points 3D de la scène filmée et le mouvement de la caméra en combinant la précision des méthodes "hors-ligne" (basées sur une optimisation globale de tous les paramètres par ajustement de faisceaux) et la vitesse de calcul des méthodes incrémentales. La nouvelle approche est considérée comme une accélération des techniques classiques de reconstruction 3D qui utilisent l'ajustement de faisceaux, permettant ainsi de traiter de longues séquences vidéos.

L'algorithme développé peut être résumé de la façon suivante : détection de points d'intérêt dans les images, mise en correspondance de ces points et sous-échantillonnage temporel de la vidéo. En effet, seul un sous-ensemble d'images dites "images clef" est sélectionné pour la reconstruction des points 3D alors que la localisation de la caméra est calculée pour chaque image. Le point clef de l'approche est l'ajustement de faisceaux local : les paramètres de la reconstruction sont affinés sur la fin de la séquence uniquement, à chaque fois qu'une image est choisie comme nouvelle image clef. La méthode, initialement prévue pour les caméras perspectives, a ensuite été généralisée de manière à rendre possible l'utilisation d'autres types de caméras, comme les caméras catadioptriques ou encore les paires rigides de caméras.

Les résultats obtenus montrent que la précision atteinte est du même ordre que celle des méthodes par optimisation globale, avec des temps de calcul très réduits, ce qui permet de viser des applications d'odométrie visuelle temps-réel pour la robotique mobile ou l'aide à la conduite en automobile.

Real-time Structure from Motion by local bundle adjustment

The Structure from Motion problem is an intense research topic in computer vision and has been the subject of much investigation. This thesis presents a method for estimating the motion of a calibrated camera and the threedimensional geometry of the filmed environment. The main idea is to take advantage of both offline methods (based on an optimization of all 3D parameters by global bundle adjustment) and fast incremental methods. The new approach may be seen as an acceleration of conventional 3D reconstruction techniques that make use of bundle adjustment, and thus enables to treat very long video sequences.

The introduced algorithm may be summarized as follows : interest points detection and matching between frames, sub-sampling of the video into "key frames", full 3D reconstruction of these key frames (3D points and camera poses), and localization of all frames. The keystone of the method is the local bundle adjustment : reconstruction parameters are refined at the end of the sequence only, for all current frame selected as key frame. This method is applied initially to a perspective camera model, then extended to a generic camera model to describe most existing kinds of cameras like catadioptric cameras or stereo rigs.

Experiments have shown that results are very similar to those obtained by methods with global optimisation, with much lower computing times. We can envisage applications like real-time visual odometry for mobile robots or car assisted driving.